TECHNICKÁ UNIVERZITA V LIBERCI

Fakulta mechatroniky a mezioborových inženýrských studií

Studijní program: M2612 – Elektrotechnika a informatika

Studijní obor: 3902T005 – Automatické řízení a inženýrská informatika

Měřící člen s CAN protokolem

Measuring unit with CAN protocol

Diplomová práce

Autor: Bc. Jiří Stejskal

Vedoucí: Ing. Josef Grosman

V Liberci 18.5.2007

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/200 o právu autorském, zejména § 60 (školní dílo).

Beru na vědomí, že TUL má právo na uzavření licenční smlouvy o využití mé diplomové práce a prohlašuji, že **souhlasím** s případným užitím mé diplomové práce (prodej, zapůjčení apod.).

Jsem si vědom toho, že užít své diplomové práce, či poskytnout licenci k jejímu využití mohu jen se souhlasem TUL, která má právo ode mne požadovat přiměřený příspěvek na úhradu nákladů, vynaložených univerzitou na vytvoření díla (až do jejich skutečné výše).

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím diplomové práce a konzultantem.

Datum:

Podpis:

Anotace:

Hlavním cílem diplomové práce je návrh a realizace měřícího členu (CAN uzlu) vybraných fyzikálních veličin, komunikujícího pomocí CAN sběrnice, který bude možno konfigurovat z PC po sériové lince

Práce seznamuje se základními principy a technickými prostředky CAN protokolu. Návrh sestává z výběru vhodného řídícího mikropočítače, řadiče a budiče CAN, volbou měřících členů, softwarového zajištění funkce mikropočítače a tvorbou konfiguračního programu pro PC.

CAN uzel byl navržen s mikropočítačem AT89C51cc03 s integrovaným CAN řadičem a budičem PCA82C250. Jako měřící člen bylo vybráno čidlo teploty a vlhkosti SHT11, k dispozici je také A/D převodník pro tři nezávislé vstupy. Řídící program uzlu byl vytvořen v jazyce C, konfigurační software pro PC v Delphi 7.0.

Funkce realizovaného CAN uzlu je v závěru ověřena pomocí komunikace s vývojovým systémem pro vytváření sběrnic s protokolem CAN.

Abstract:

The major goal of diploma work is design and realization of measuring unit (CAN node) of chosen physical values, which communicates via CAN bus and will be possible to configure it from PC via serial port.

This work apprises fundamental principles, technical instruments and applications of CAN protocol. The proposal consists of comparison and selection acceptable microcontrollers and CAN transceiver, measuring units, software support for microcontroller and of creation of configuration program for PC.

CAN node was designed with microcontroller AT89C51c03 with integrated CAN transceiver and bus driver PCA82C250. As a measuring unit was chosen sensor of temperature and humidity SHT11, available is A/D converter too. Main program was created in C language, configuration software in Delphi 7.

The function of this device is verified by using of communication with the development system with CAN in the end.

Obsah

Termíny a zkratky

ISO/OSI	zkratka mezinárodní organizace pro normalizaci „International Standards Organization / Open System Interconnection“
NRZ	způsob kódování dat bez návratu k nule „Non Return to Zero“
ISP	způsob programování mikroprocesoru bez nutnosti vyjmutí ze zařízení „In System programming“
Bootloader	zaváděcí program mikroprocesoru umožňující ISP
UART	sériové rozhraní mikroprocesoru „Universal Asynchronous Receiver and Transmitter“
PACKET	datová jednotka užívaná při komunikaci mezi zařízeními

Úvod

Sběrnice CAN je sériový komunikační protokol umožňující řízení systémů v reálném čase s vysokou mírou zabezpečení. Byl vyvinut v 80. letech společností Bosch a původně byl určen ke komunikaci mezi automobilovými řídícími jednotkami se záměrem úspory kabeláže a zabezpečení přenosu informací. První nasazení CAN v automobilech proběhlo v roce 1992 v automobilce Mercedes Benz a od roku 1996 je využíván také automobilkou Škoda Auto. Přestože byl CAN navržen pro automobilový průmysl, nachází tento protokol díky svým vlastnostem a podpoře předních výrobců integrovaných obvodů čím dál větší uplatnění v průmyslových aplikacích, jako řízení vzdálených zařízení, předávání informací v technologii, či v systémech inteligentních budov. V současné době má protokol CAN své pevné místo mezi ostatními fieldbusy, které vytlačily většinu analogových informačních rozvodů a je definován normou ISO 11898.

Právě tento, dá se říct moderní, komunikační standard je tématem této diplomové práce, jejímž cílem je realizace zařízení (CAN uzlu), které umožní sběr dat z vybraných snímačů a bude komunikovat s ostatními uzly připojenými ke sběrnici. Toto zařízení by mělo být možné použít v průmyslu mimo jiné také jako např. univerzální měřící prvek nějaké jiné, rozsáhlejší technologie. Vzhledem k účelu tohoto zařízení byl při návrhu kladen důraz na kompaktnost, minimalizaci rozměrů a nízkou energetickou náročnost z důvodu možnosti napájení z baterií.

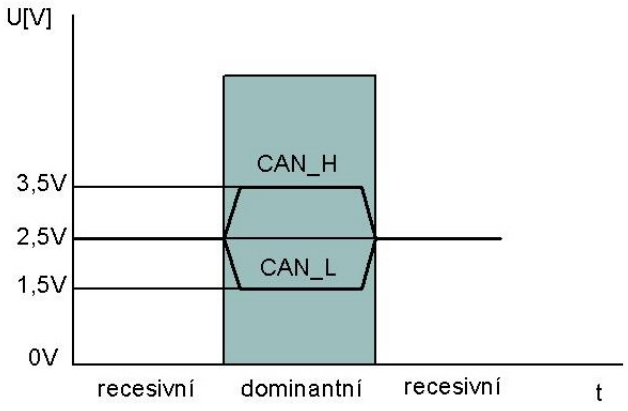
První část této práce je věnována samotnému CAN protokolu. Popisuje jeho základní principy a vlastnosti a první dvě vrstvy z referenčního modelu ISO/OSI, fyzickou a linkovou, které jsou standardizované normou. V druhé kapitole jsou popsány základní stanovené požadavky na navrhované zařízení, podle kterých byly následně vybrány jednotlivé komponenty. Je zde popsáno celkové obvodové řešení včetně návrhu desky plošných spojů. Třetí kapitola je zamřena na vývoj software řídícího mikroprocesoru a vývoj aplikace pro PC, pomocí které je možné konfigurovat některé významné vlastnosti CAN uzlu, jako je přenosová rychlost CAN sběrnice, identifikátory zpráv, případně volba veličin, které se budou vysílat. V závěru práce jsou shrnuty vlastnosti navrženého CAN uzlu jako celku a zhodnoceno výsledné řešení úkolu.

1 CAN – Control Area Network

CAN byl navržen pro přenosové rychlosti do 1Mb/s s velkým zabezpečením dat proti chybám. Jedná se o protokol typu multi-master, to znamená, že síť není nutné řídit z jednoho nadřazeného uzlu, což přináší zjednodušení řízení a zvyšuje spolehlivost sítě. Využívá se kódování NRZ s vkládáním bitů (blíže v kapitole 1.2.2) a pro řízení přístupu k médiu je použita sběrnice s náhodným přístupem, která řeší kolize na základě prioritního rozhodování, tzv. arbitráže (viz kapitola 1.2.1). Komunikace probíhá na základě zpráv, přičemž tyto neobsahují žádné informace, komu jsou určeny, ale pomocí filtrování zpráv je zajištěno, aby připojený uzel přijímal pouze ty zprávy, které jsou mu určeny. Všechny tyto základní vlastnosti jsou standardizovány od roku 1993 dle normy ISO 11898, která zachycuje dvě nejnižší vrstvy, fyzickou a linkovou, z referenčního modelu pro síťové protokoly ISO/OSI. Vyšší vrstvy modelu nejsou definovány. V některých případech se využívá vrstva aplikační, pro kterou sice prozatím neexistuje žádná norma upravující její vlastnosti, ale existuje několik vzájemně nekompatibilních standardů, sdružovaných uživatelskou organizací CiA. Podporovány jsou např. CANopen, či DeviceNet.

1.1 Fyzická vrstva

Fyzická vrstva je velkou předností protokolu CAN. Je definována jako sériová se dvěmi navzájem komplementárními stavy *recessive* a *dominant*. Tyto stavy jdou v podstatě jakýmsi ekvivalentem logických úrovní.



Obr. 1.1: Logické úrovně vodičů CAN sběrnice

Základním požadavkem na přenosové médium je umožnění realizace logického součinu, čímž jsou určeny jednoznačná pravidla. Pokud všechna zařízení připojená ke sběrnici vysílají hodnotu *recessive*, je celá sběrnice ve stavu *recessive*. Pokud alespoň jedno zařízení vysílá hodnotu *dominant*, je celá sběrnice ve stavu *dominant*.

Nejčastěji se využívá dvouvodičová diferenciální sběrnice s vodiči označenými CAN_H a CAN_L na koncích zakončenými rezistory o hodnotě 120Ω. Stav sběrnice je dán rozdílovým napětím mezi těmito vodiči.

Nominální hodnoty logických úrovní jsou znázorněny na obr. 1.1, str. 8. V *recesivní* úrovni jsou potenciály obou vodičů shodné a jako *recesivní* úroveň je vyhodnocen rozdíl mezi oběma vodiči menší než 0,5V. *Dominantní* úrovni odpovídá rozdíl mezi vodiči větší než 0,9V, přičemž nominální hodnoty napětí *dominantního* stavu jsou 3,5V pro CAN_H a 1,5V pro CAN_L.

Jednotlivá zařízení připojovaná na sběrnici jsou označovány jako uzly a připojují se pomocí odboček (většinou konektorem CANON-9M), čímž získají přímý přístup k ostatním uzlům na sběrnici. Může jich být teoreticky připojeno neomezené množství, avšak s ohledem na zatížení sběrnice je jich doporučováno maximálně 30.

Norma uvádí pro maximální přenosovou rychlost 1Mbit/s maximální délku sběrnice 40m. Pro nižší přenosové rychlosti je maximální délka sběrnice větší. Orientační hodnoty jsou na obr. 1.2.

PŘENOSOVÁ RYCHLOST	MAXIMÁLNÍ DÉLKA SBĚRNICE
1Mbit/s	40m
500kbit/s	112m
300kbit/s	200m
100kbit/s	640m
50kbit/s	1340m
20kbit/s	2600m
10kbit/s	5200m

Obr. 1.2: Maximální délka CAN sběrnice

1.2 Linková vrstva

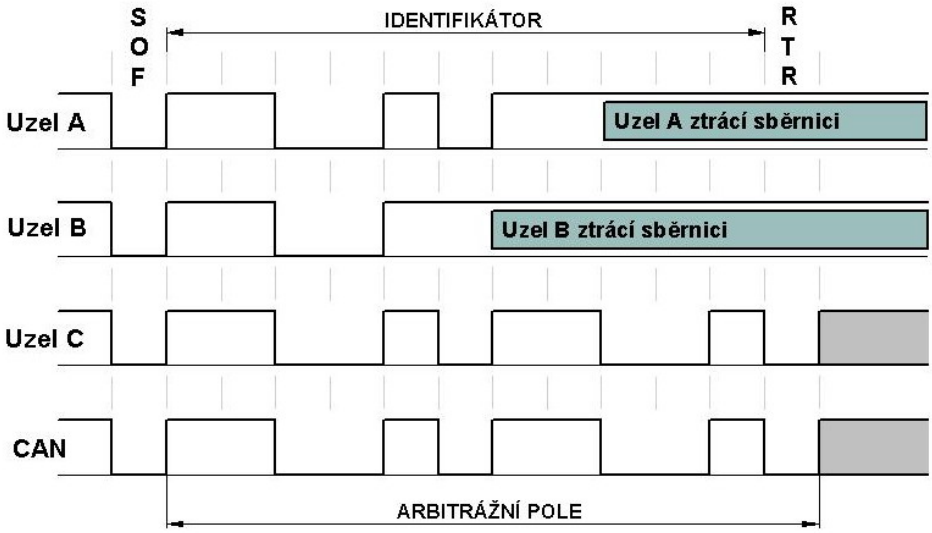
Linková vrstva protokolu CAN je tvořena dvěma podvrstvami MAC a LLC. První z nich reprezentuje jádro protokolu a zajišťuje kódování dat, vkládání doplňkových bitů, řízení přístupu jednotlivých uzlů ke sběrnici s rozlišením priority

zpráv, detekci chyb a potvrzování přijatých zpráv. LLC vrstva má za úkol zejména filtrovat přijaté zprávy a hlásit přetížení uzlu.

1.2.1 Řízení přístupu ke sběrnici

Vzhledem k tomu, že se jedná o multi-master síť, je přístup na sběrnici umožněn libovolnému připojenému uzlu, pokud je připraven vysílat a síť je v klidovém stavu. Tato metoda přístupu je náhodná a síť získá ten uzel, který začne vysílat dříve. Ostatní uzly jsou ve stavu příjmu a mohou začít vysílat až v okamžiku, kdy je zpráva odvysílána. Výjimkou jsou tzv. chybové rámce, které jsou odvysílány ihned po detekci chyby kterýmkoliv uzlem.

V případě, že začne vysílat více uzlů najednou, provádí se tzv. arbitráž. Všechny zúčastněné uzly vysílají své zprávy a zároveň je porovnávají s hodnotou čtenou ze sběrnice. Pokud se tato data rovnají, kolize nevzniká a uzly pokračují ve vysílání. V případě, že stanice vyslala recesivní bit a přijala bit dominantní, začne namísto zprávy vysílat recesivní hodnotu a než se sběrnice uvolní, chová se jako přijímač. Příklad arbitrace třech uzlů je na obr.1.3.



Obr. 1.3: Arbitráž o přístup na sběrnici

Arbitrace se provádí během arbitrážního pole zprávy, které se ve standardním formátu skládá z tzv. identifikátoru zprávy a bitu RTR (viz dále kapitola 1.3). Arbitrací je zajištěno, že přednost dostane zpráva s nižším identifikátorem, resp. zpráva datová.

1.2.2 Detekce a signalizace chyb a zabezpečení dat

Při přenosu zprávy může například vlivem rušení dojít k jejímu poškození. Toto poškození musí být detekováno a přenášená zpráva stornována. Nebo pokud nějaký uzel opakovaně generuje chyby, kterými zahltí sběrnici, je potřebné nějakým způsobem řídit přístup poškozených uzlů ke sběrnici. K tomuto slouží následující prostředky:

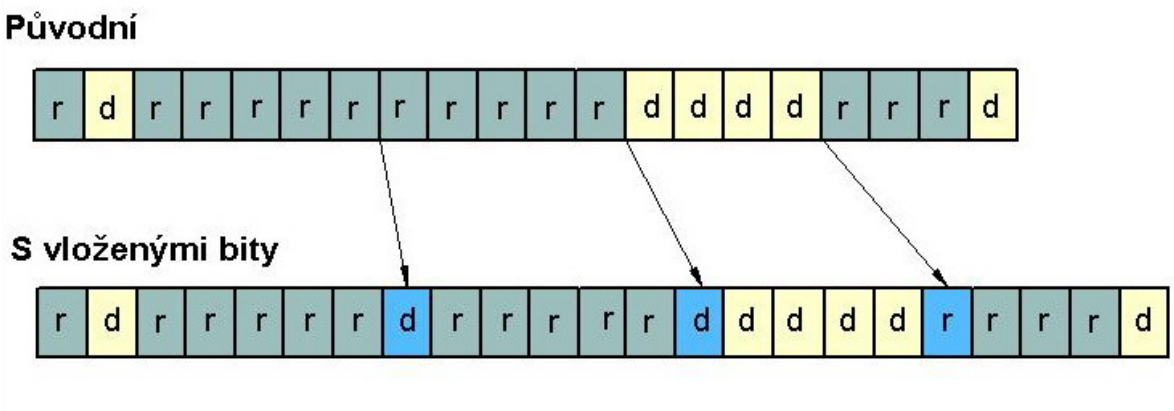
Kontrola odeslaných dat (Monitoring):

Již zmíněná metoda spočívá v porovnávání odesílaných bitů s bity na sběrnici. Pokud jsou bity rozdílné, mohou nastat dvě situace:

1. Jestliže rozdíl nastane během arbitrážního pole, znamená to, že jiný uzel vysílá zprávu s nižší prioritou a uzel detekující rozdíl se přepne do stavu příjmu.
2. Jestliže rozdíl nastane mimo arbitrážní pole, uzel generuje *chyba bitu.*

Vkládání bitů (Bit stuffing):

Při použití NZR (Non Return to Zero) bitového kódování zůstává hodnota bitu po čas jeho trvání stejná a jednotlivé bity jsou rozlišovány pomocí časových slotů a je nutná synchronizace. Může se stát, že po sobě bude následovat více bitů stejné úrovně a to může ztížit synchronizaci. Proto je po pěti po sobě jdoucích bitech stejné úrovně vložen bit úrovně opačné, který je na straně příjmu odstraněn. Oblast, kam se vkládají bity, zahrnuje začátek zprávy SOF, arbitrážní, řídící a datové pole a kontrolní součet CRC (popis polí viz kapitola 1.3). Pokud je toto pravidlo porušeno, je generována *chyba vkládání bitů* a zpráva je ve všech přijímačích stornována a znovu opakována.



Obr.1.4: Příklad vkládání bitů do zprávy

Kontrola pomocí CRC kódu:

Na konci každé zprávy je uveden 15-ti bitový CRC kód, který je generován ze všech předchozích bitů, které již byly odvysílány. Po přijetí je CRC kód znovu spočítán a pokud se liší od přijatého CRC kódu, kterýkoliv uzel na sběrnici vysílá *chybu CRC.*

Kontrola zprávy (Message Frame Check):

Každá zpráva se kontroluje podle specifikace rámce a pokud je na pozici nějakého bitu zjištěna nepovolená hodnota, je generována *chyba rámce.*

Potvrzení přijetí zprávy (Acknowledge):

Je-li zpráva v pořádku přijata libovolným uzlem, je to potvrzeno změnou hodnoty bitu zprávy vyhrazeného pro odpověď (ACK bit). Vysílající uzel vždy na tomto bitu vysílá recesivní úroveň. Je-li detekována dominant úroveň, je přenos považován za úspěšný. Potvrzení přijetí je prováděno všemi uzly připojenými na sběrnici bez ohledu na nastavené filtrování zpráv.

Každý uzel obsahuje dvě interní počítadla chyb, udávající počet chyb příjmu a vysílání. Pokud uzel generuje příliš mnoho chyb, je automaticky odpojen. Obecně je podle počtu chyb je uzel přepnut do jednoho z následujících tří stavů:

- Aktivní – uzel se aktivně podílí na komunikaci a v případě, že detekují libovolnou chybu v právě přenášené zprávě, vysílají na sběrnici aktivní příznak chyby, který je tvořen šesti po sobě jdoucími bity dominant, čímž se poruší pravidlo vkládání bitů.
- Pasivní – uzel se také podílí na komunikaci, ale v případě detekce chyby vysílají pouze pasivní příznak chyby, což je šest po sobě jdoucích bitů recessive
- Odpojené – tyto uzly se žádným způsobem nepodílí na komunikaci, jejich výstupní budiče jsou odpojené.

1.3 Základní typy zpráv

Specifikace protokolu CAN 2.0 definuje čtyři druhy zpráv. Datovou zprávu (Data Frame), zprávu žádosti o data (Remote Frame), chybovou zprávu (Error Frame) a zprávu o přetížení (Overload Frame).

CAN 2.0 rozlišuje navíc dvě verze datových zpráv. První typ je definován specifikací 2.0A a označován jako standardní formát zprávy (Standard Frame) s 11 bitovým identifikátorem. Specifikace 2.0B definuje kromě standardního formátu zprávy navíc rozšířený formát zprávy (Extended Frame) s 29 bitovým identifikátorem.

1.3.1 Datová zpráva dle specifikace CAN 2.0A

Datová zpráva (Data Frame) je základním komunikačním prvkem, která umožňuje přenést 0-8 datových bajtů. Pro jednoduché zprávy, resp. příkazy může být datová oblast prázdná a příkaz může být zakódován do identifikátoru. Struktura standardního datového rámce dle specifikace CAN 2.0A je na obr. 1.5.

Obr.1.5: Struktura standardního datového rámce dle CAN 2.0A

Význam jednotlivých bitů standardního rámce podle specifikace 2.0A:

- SOF (Start Of Frame) – 1 bit dominant, značí začátek zprávy a slouží k synchronizaci všech přijímačů s vysílačem na začátku vysílání
- Arbitrážní pole (Arbitration Field) – určuje prioritu přenášené zprávy
  - Identifikátor zprávy – 11 bitů udávajících význam přenášené zprávy
  - RTR (Remote Request) – 1 bit příznaku, zda se jedná o datovou zprávu, nebo o žádost o data. V datové zprávě je tento bit *dominant*
- Řídící pole (Control Field) – 6 bitů
  - R0, R1- rezervované bity
  - Délka dat DLC (Data Length Code) – 4 bity udávající počet datových bajtů
- Datové pole (Data Field) – 0 až 8 datových bajtů zprávy

- Zabezpečovací pole (CRC Field) – 16 bitů
  - CRC – 15 bitů CRC kódu
  - ERC – 1 bit *recessive* ukončující CRC pole
- Potvrzení (ACK Field) – 2 bity
  - ACK – 1 bit potvrzení příjmu vysílaný *recessive* a přepsán na *dominant* přijímacími uzly, pokud je přenos v pořádku
  - ACD – 1 bit *recessive* oddělující potvrzení
- Konec rámce (End of Frame) – 7 bitů *recessive* ukončujících zprávu
- Mezera mezi zprávami (Interframe Space) – 3 bity *recessive* oddělující zprávy

1.3.2 Datová zpráva dle specifikace CAN 2.0B

Specifikace CAN 2.0B definuje dva formáty datové zprávy: standardní a rozšířený. Standardní zpráva je z důvodu kompatibility převzata ze specifikace CAN 2.0A a liší se pouze využitím bitu R1 jako bit IDE informující o tom, zda se jedná o rámec standardní, nebo rozšířený. Pro standardní rámec je tento bit *dominantní*. Jestliže je tento bit *recesivní*, bude následovat dalších 18 bitů identifikátoru a jde o rozšířený datový rámec, jehož začátek je na obr. 1.6.

Obr.1.6: Struktura rozšířeného datového rámce dle CAN 2.0B

Identifikátor v rozšířeném rámci (Extended Frame) je rozdělen na pole o 11 a 18 bitech. Bit RTR je oproti standardnímu formátu nahrazen bitem SRR (Substitute Remote Request), který má vždy hodnotu *recessive*. To zajišťuje, že při vzájemné kolizi standardního a rozšířeného formátu zprávy se stejným 11 bitovým identifikátorem získal přednost standardní rámec. Bit IDE (Identifer Extended) má vždy hodnotu *recessive*. Bit RTR je přesunut za druhou část identifikátoru.

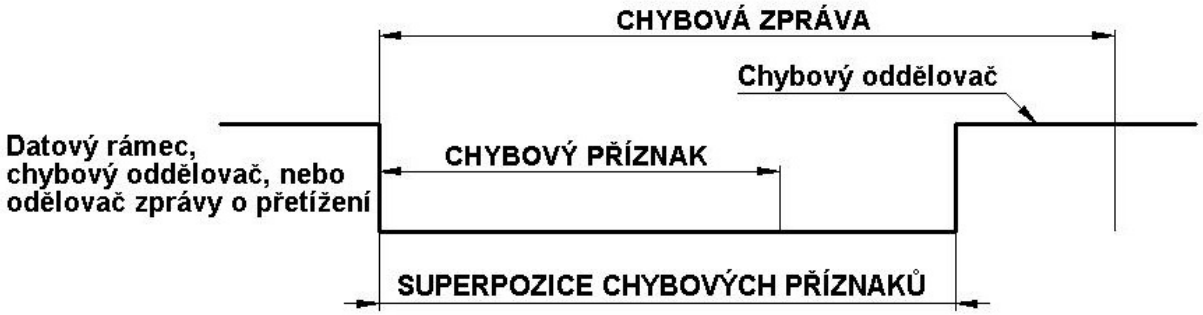
1.3.3 Zpráva žádosti o data

Cílová stanice může vyslat požadavek na data prostřednictvím zprávy žádosti o data (Remote Frame) s identifikátorem, který odpovídá identifikátoru požadovaného datového rámce. Zdrojová stanice poté vyšle datový rámec jako odpověď.

Mezi datovým rámcem a žádostí o data jsou dva rozdíly. V žádosti o data je bit RTR vysílán jako *recesivní* a neobsahuje datovou oblast. V případě, že začnou být současně vysílány datový rámec a žádost o data, dostane v arbitráži přednost datový rámec kvůli *dominantnímu* bitu RTR.

1.3.4 Chybová zpráva

Chybová zpráva (Error Frame) slouží k signalizaci chyb na sběrnici. Jakmile libovolný uzel na sběrnici detekuje v přenášené zprávě chybu, vygeneruje ihned na sběrnici chybový rámec. Podle toho, v jakém stavu pro hlášení chyb se uzel, který zjistil chybu, právě nachází, generuje na sběrnici buď aktivní (šest bitů *dominant*), nebo pasivní (šest bitů *recessive*) příznak chyby. Hlášení chyb je pak indikováno superpozicí všech chybových příznaků, které vysílají jednotlivé uzly. Délka tohoto úseku může být minimálně 6 a maximálně 12 bitů.

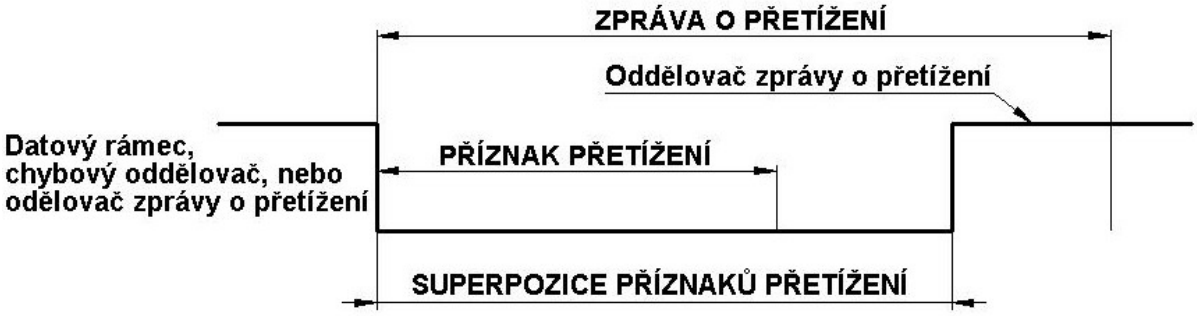


Obr. 1.7: Chybová zpráva

Po vysílání chybového příznaku vysílá každá stanice na sběrnici bity *recessive*. Zároveň detekuje stav sběrnice a jakmile najde první bit na sběrnici ve stavu *recessive*, vysílá dalších sedm bitů *recessive*, které plní funkci oddělovače chyb.

1.3.5 Zpráva o přetížení

Zpráva o přetížení (Overload Frame) slouží k oddálení vysílání další datové zprávy nebo žádosti o data. Zpravidla tento způsob využívají zařízení, která nejsou schopna kvůli svému vytížení přijímat a zpracovávat další zprávy. Struktura zprávy o přetížení (viz obr. 1.8) je podobná zprávě chybové, ale její vysílání může být zahájeno až po konci zprávy, oddělovače chyb, nebo předcházejícího oddělovače zpráv přetížení.



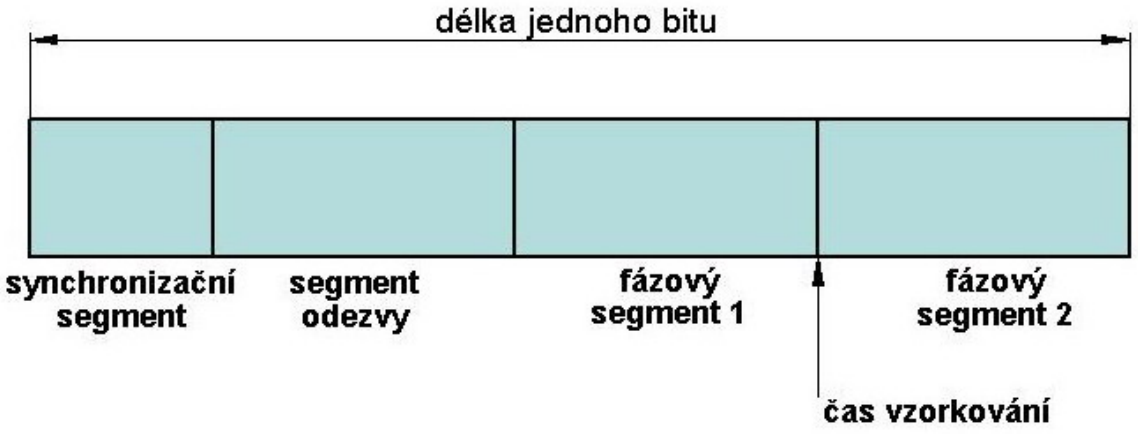
Obr.1.8: Zpráva o přetížení

Zpráva o přetížení je složena z příznaku přetížení (šest bitů *dominant*) a případné superpozice všech příznaků přetížení, pokud jsou generovány více uzly současně. Za příznakem přetížení následuje dalších sedm bitů *recessive*, které tvoří oddělovač zprávy o přetížení.

1.4 Časování bitů a synchronizace

Sběrnice CAN využívá k časování hodinový signál odvozený od frekvence oscilátoru. Vzhledem k tomu, že každý uzel má svůj vlastní oscilátor, může dojít k fázovému posunu. Tento hodinový signál je v předděliči (Baud Rate Prescaler - BRP) vydělen zvolenou celočíselnou hodnotou a výstupem je signál s periodou t_q. Tato perioda se nazývá *časové kvantum* (time quantum) a je výchozí pro stanovení přenosové rychlosti sběrnice.

Délka jednoho bitu se skládá ze čtyř nepřekrývajících se segmentů, z nichž každý je složen z celočíselného počtu časových kvant t_q. Celkový počet časových kvant musí být 8 až 25.



Obr. 1.9: Složení délky jednoho bitu

- synchronizační segment (Sync segment) – je dlouhý $1t_q$ a slouží pro synchronizaci přijímačů s vysílači
- segment odezvy (Propagation segment PRS) – je programovatelně dlouhý 1 až $8t_q$ a je použit pro kompenzaci časových zpoždění na sběrnici
- fázový segment 1 (Phase segment 1 PHS1) – je programovatelný v délce 1 až $8t_q$ a je použit pro kompenzaci fázového rozdílu mezi přijímaným signálem a vnitřním časovačem. Segment může být prodloužen během resynchronizace
- fázový segment 2 (Phase segment 2 PHS2) – je roven maximální hodnotě z fázového segmentu 1 a dobou zpracování informace (Information processing time), který je roven $2t_q$
- čas vzorkování (Sample Time) - je to programovatelný okamžik (v 60-80% délky bitu), ve kterém se přečte hodnota ze sběrnice a vezme se jako platný údaj. Umožňuje optimalizovat vzorkování s ohledem na délku sběrnice a kvalitu hran.

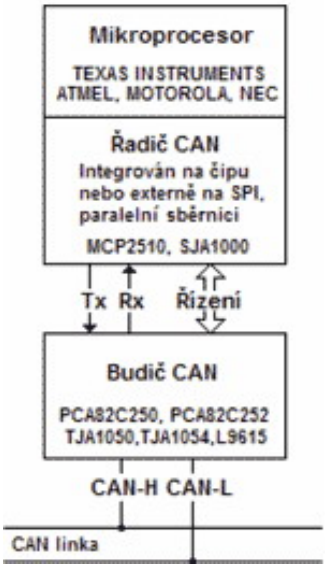
Synchronizace probíhá při každé změně úrovně příchozího signálu, přičemž příchod hrany se očekává v průběhu synchronizačního segmentu. Pokud hrana přijde později, fázový rozdíl je kompenzován prodloužením fázového segmentu 1 maximálně o programovatelnou hodnotu proměnné SJW+1 (minimum při SJW = 0 je $1t_q$), která udává celočíselný násobek časového kvanta t_q . Přijde-li hrana vstupního signálu dříve, dojde ke zkrácení fázového segmentu 2 opět maximálně o SJW+1. Vkládáním bitů je zaručena synchronizace maximálně každých 5 bitů.

2 Návrh hardware

V této kapitole jsou popsány požadavky na zařízení jako celek a z toho vyplívající požadavky na hardware. Popsána je volba jednotlivých komponent, např. mikroprocesoru, budiče CAN sběrnice či výběr snímačů a jsou také popsána jednotlivá obvodová řešení. Při návrhu byl kladen důraz na snadnou konstrukci, kompaktnost, malé rozměry, nízkou spotřebu umožňující dlouhodobé napájení z baterie a řešení umožňující snadnou a častou změnu programu pomocí ISP (In System Programming).

2.1 Kriteria výběru komponent a požadavky na HW

Hlavním rysem navrhovaného zařízení je schopnost komunikovat po CAN sběrnici, proto jsem při návrhu postupoval od tohoto požadavku. Struktura zařízení přistupujícího na sběrnici CAN je na obr. 2.1. Mikroprocesor obsahuje řídící program, který pomocí CAN řadiče vytváří samotný protokol sběrnice. CAN řadič bývá díky podpoře výrobců, jako například Infineon, Motorola, Microchip, Atmel, Philips, atd. již integrován do některých mikroprocesorů, což usnadňuje hardwarový i softwarový návrh a šetří místo na desce plošného spoje. Externí CAN řadiče se používají v případě potřeby zvýšení počtu dostupných CAN sběrnic a komunikují s procesorem pomocí sériové, nebo paralelní sběrnice.



Obr. 2.1: Struktura CAN zařízení [5]

K fyzickému připojení řadiče ke sběrnici slouží CAN budič. Ty se rozlišují podle počtu vodičů a maximální přenosové rychlosti na low speed a high speed. Pro návrh CAN komunikace jsem zvolil cestu integrovaného řadiče CAN v procesoru s podporou standardu CAN 2.0B spolu s high speed budičem do rychlosti 1Mbit/s.

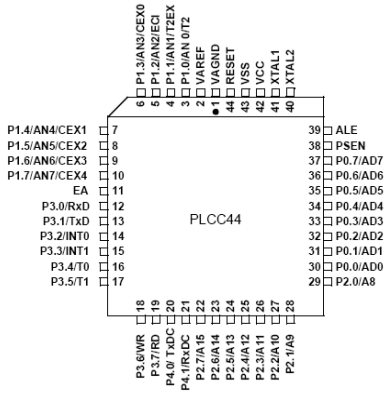
Dalším důležitým požadavkem je konfigurace navrhovaného zařízení po sériové lince RS232C a uchování konfiguračních hodnot v paměti EEPROM, což vyžaduje přítomnost obvodů umožňujících tyto funkce. Posledním, ale neméně důležitým prvkem je A/D převodník, protože modul by měl umožňovat měřit i fyzikální veličiny z čidel s analogovým výstupem.

2.2 Popis hlavních komponent

2.2.1 Procesor

Jak již bylo zmíněno, procesor by měl v sobě integrovat CAN řadič dle CAN2.0B a umožňovat snadnou změnu programu pomocí ISP. Přítomnost CAN řadiče a ISP dnes není nic neobvyklého, téměř každý výrobce toto řešení nabízí. Podpora rozhraní RS232 procesorem je dnes standardem a proto jsem začal přemýšlet o řešení s integrovanou EEPROM pamětí a A/D převodníkem, čímž by téměř všechny prostředky pro celé zařízení byly obsaženy v jednom čipu, což by zaručilo kompaktnost, jednoduchost návrhu a toto řešení by se pozitivně odrazilo i v celkové spotřebě zařízení.

Rozhodl jsem se pro procesor firmy ATMEL AT89C51cc03U-SLSIM v pouzdře PLCC44, které umožní osazení do patice a se sériovým bootloaderem pro ISP (viz popis ISP v kapitole 3.1).



Obr. 2.2: Procesor AT89C51cc03 v pouzdru PLCC44 [9]

Základní využívané vlastnosti mikroprocesoru:

- 8051 architektura
- 256 bajtů interní paměti RAM, 2048 bajtů interní ERAM
- 64k bajtů paměti FLASH pro program, 2048 bajtů interní EEPROM
- 3 16 bitové čítače/časovače
- 14 zdrojů přerušení se 4 úrovněmi priority
- plně duplexní UART kompatibilní s 80C51
- 10 bitový A/D převodník s 8 multiplexovanými vstupy
- ISP pomocí sériového bootloaderu
- CAN řadič kompatibilní s CAN 2.0A a 2.0B s 15 objekty pro zprávy
- Napájení +5V, napájecí proud Icc = (0,4 . frekvence (Mhz) + 8) mA

2.2.2 CAN budič

CAN budič zajišťuje převod dat z procesoru na CAN sběrnici a opačně. Kromě požadavku na high speed režim přenosu a napájení 5V jsem bral ohled také na možnost uvedení budiče do režimu nízké spotřeby v případě nulové komunikace. Pro realizaci jsem vybral rozšířený obvod PCA82C250 od firmy Philips, který je také osazen na vývojových prostředcích pro CAN v učebně KSI. Pro snadné osazení jsem použil pouzdro DIP 8. Popis vývodů je na obr. 2.3.

VÝVOD	PIN	POPIS
TxD	1	data k vyslání (od procesoru)
GND	2	zem
Vcc	3	napájecí napětí
RxD	4	přijímaná data (k procesoru)
Vref	5	výstup referenčního napětí (0,5Ucc)
CANH	6	vstup/výstup CAN sběrnice
CANL	7	vstup/výstup CAN sběrnice
Rs	8	řízení režimu

Obr. 2.3: Popis vývodů PCA82C250

Tento obvod podporuje tři režimy přenosu, které jsou řízeny logickou úrovní na vstupu Rs budiče. Jsou to režimy high speed, slope control a low power. Připojením vstupu Rs na GND je budič v režimu high speed, ve kterém je přechod mezi úrovněmi CAN_H a CAN_LOW uskutečněn tak rychle, jak dovolí hardware. V režimu slope control je kontrolována rychlost náběžné, resp. sestupné hrany. Hodnoty jsou dány rezistorem připojeným mezi Rs a GND.

Připojením vstupu Rs na Ucc se budič uvede do režimu low power, kdy je vysílač vypnut a přijímač přepnut do režimu nízkého příkonu. Jakmile je na přijímači detekována dominantní úroveň sběrnice, budič změní na vodiči RxDC směrem k procesoru logickou úroveň a procesor může reagovat přepnutím vstupu Rs budiče a uvést ho do standardního operačního režimu. Prodleva do uvedení do tohoto režimu je dle katalogu 20 μs. Nevýhodou je ztráta první zprávy z důvodu přechodu do standardního režimu. Proudový odběr tohoto budiče v high speed režimu je < 20 mA a 170 μA v režimu low power. Výstup je vyveden standardním konektorem D-SUB9F.

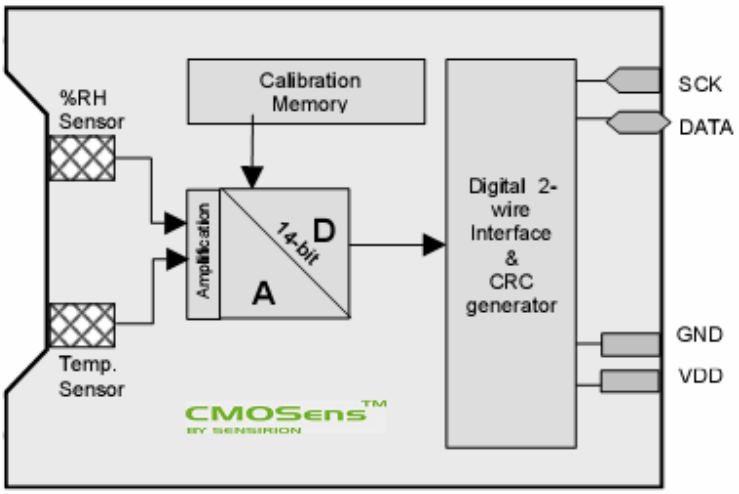
2.2.3 Senzor SHT11

Senzor SHT11 je od firmy Sensirion a integruje v jednom čipu snímače teploty a relativní vlhkosti, přičemž jejich výstupní hodnoty jsou přímo v čipu převedeny do digitální podoby a následně dále zpracovávány. Kromě uvedených snímačů senzor obsahuje 14 bitový A/D převodník, kalibrační paměť (OTP) a 2-drátové rozhraní pro komunikaci s okolím, sestávající ze signálu hodin SCK a obousměrného datového signálu DATA. Kromě těchto dvou signálů stačí pro zajištění funkčnosti připojit již jen piny napájení. Tento senzor (oproti ostatním senzorům vlhkosti) není nutné před použitím kalibrovat, protože jsou již kalibrovány ve vlhkostní komoře při výrobě a kalibrační koeficienty jsou uloženy v kalibrační paměti. Tyto koeficienty jsou použity interně během měření pro přepočítání údajů ze senzorů.

Senzor je dodáván v poměrně specifickém pouzdře pro povrchovou montáž LCC (Leadless Chip Carrier), nebo „kolíkovém“ 4-pinovém single-in-line pouzdře.

Integrovaný A/D převodník umožňuje 14 bitové rozlišení naměřených hodnot, nicméně v tomto rozlišení je měřena pouze teplota z přesnějšího snímače teploty, vlhkosti vyhoví 12 bitové rozlišení. Tyto hodnoty mohou být pomocí konfigurace registru redukovány na 12 a 8 bitů při požadavku velmi rychlých měření, nebo pro

snížení spotřeby. Kromě výše uvedených vlastností bych vyzdvihnul ještě velmi malou spotřebu s automatickým řízením snížené spotřeby, malé rozměry a výbornou dlouhodobou stabilitu. Blokové schéma SHT11 je na obr. 2.4.



Obr. 2.4: Blokové schéma senzoru SHT11 [12]

Základní technické údaje:

- napájecí napětí 2,4 – 5,5V
- průměrná spotřeba 28μA při měření s rozlišením 12 bitů jednou za sekundu, v režimu snížené spotřeby 1μA
- rozsah měření teploty (-40 až 124) °C
- přesnost měření teploty ±0,4°C při 25°C, ±1°C v rozsahu (0 až 50) °C a ±2°C v rozsahu (-40 až 90) °C
- rozsah měření relativní vlhkosti (0 až 100) %RH
- přesnost měření relativní vlhkosti ±3 %RH v rozsahu (20 až 80) %RH, ±5%RH jinak

Převod digitálních hodnot na fyzikální veličiny

Výstup čidla relativní vlhkosti není lineární a je proto vhodné tuto nelinearitu kompenzovat. Dále je také možné kompenzovat naměřenou hodnotu vlhkosti pomocí teploty při měření mimo teplotu 25°C.

Aplikační list výrobce doporučuje následující vztahy a konstanty:

Kompenzace vlhkosti vlivem nelinearity čidla:

$$RH_{LINEAR} = c_1 + c_2 \cdot SO_{RH} + c_3 \cdot SO_{RH}^2 \qquad [\%] \qquad (2.1)$$

SO_{RH} *[%]* … digitální hodnota vlhkosti z výstupu čidla

SO_{RH}	c_1	c_2	c_3
12 bit	-4	0,0405	$-2{,}8 \cdot 10^{-6}$
8 bit	-4	0,648	$-7{,}2 \cdot 10^{-3}$

Obr. 2.5: Konstanty pro linearizaci

Teplotní kompenzace vlhkosti:

$$RH_{TRUE} = (T - 25) \cdot (t_1 + t_2 \cdot SO_{RH}) + RH_{LINEAR} \qquad [\%] \qquad (2.2)$$

T *[°C]* … vypočtená hodnota teploty (viz. níže)

SO_{RH} *[%]* … digitální hodnota vlhkosti z výstupu čidla

RH_{LINEAR} *[%]* … linearizovaná hodnota vlhkosti

SO_{RH}	t_1	t_2
12 bit	0,01	0,00008
8 bit	0,01	0,00128

Obr. 2.6: Konstanty pro teplotní kompenzaci

Výpočet teploty z digitální hodnoty:

Teplotní čidlo je více lineární, proto je výpočet hodnoty teploty z digitálního vyjádření jednodušší.

$$T = d_1 + d_2 \cdot SO_T \qquad [°C] \qquad (2.3)$$

SO_T *[°C]* … digitální hodnota teploty z výstupu čidla

SO_T	d_1 (°C)	d_1 (°F)	SO_T	d_2 (°C)	d_2 (°F)
5V	-40,00	-40,00	12 bit	0,01	0,018
4V	-39,75	-39,55	8 bit	0,04	0,072
3,5V	-39,66	-39,35			
3V	-39,60	-39,28			
2,5V	-39,55	-39,23			

Obr. 2.7: Konstanty pro výpočet teploty

Výpočet rosného bodu

Rosný bod (Dew-Point) je možné vypočítat celkem snadno z relativní vlhkosti a teploty podle následujícího vztahu:

$$DP(T,RH) = \frac{\lambda \cdot \left(\ln\left(\frac{RH_{TRUE}}{100} \right) + \frac{\beta \cdot T}{\lambda + T} \right)}{\beta - \left(\ln\left(\frac{RH_{TRUE}}{100} \right) + \frac{\beta \cdot T}{\lambda + T} \right)} \quad [°C] \qquad (2.4)$$

RH_{TRUE} *[%]* … kompenzovaná hodnota vlhkosti

T [°C] … vypočtená hodnota teploty, konstanty α = 6,112, β = 17,62 a λ=243,12.

Status registr

Status registr senzoru SHT11 (viz obr. 2.8) umožňuje využít několika funkcí integrovaných do čidla.

BIT	POPIS	DEFAULT
7	rezervován	0
6	END OF BATERY "1" - VCC < 2,45V "0" - VCC > 2,45V	X
5	rezervován	0
4	rezervován	0
3	rezervován	0
2	HEATER	0 - vypnuto
1	NO RELOAD OTP	0 - reload
0	RESOLUTION "1" - 8bit RH a 12bit teplota "0" - 12bit RH a 14bit teplota	0 – 12/14

Obr. 2.8: Význam bitů status registru

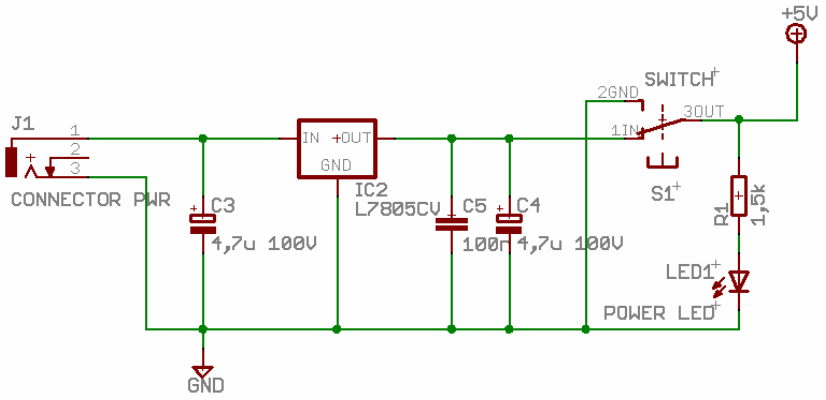
- END OF BATTERY – tento bit detekuje pokles napájecího napětí VCC pod 2,45V
- HEATER – funkce vyhřívání, která zvýší teplotu o 5°C při růstu spotřeby o 8mA/5V a může být využito pro kontrolu správné funkce teplotního senzoru, případně může zabránit kondenzaci vlhkosti v prostředí s vysokou vzdušnou vlhkostí
- NO RELOAD OTP – vypnutí nahráváni kalibračních dat z OTP paměti do paměti pracovní při každém měření. Důsledkem je zrychlení měření o 8ms.
- MEASURE RESOLUTION – rozlišení měření. Základní je 14 bitů pro teplotu a 12 bitů pro vlhkost. Nastavením lze rozlišení redukovat na 12, resp. 8 bitů.

2.3 Jednotlivá obvodová řešení

2.3.1 Napájecí obvod

K napájení celé desky plošných spojů jsem použil napájecí obvod sestavený pomocí stabilizátoru L7805CV (ST Microelectronics) v pouzdru TO-220. Výstupní napětí je v rozmezí 4,8 až 5,2V, což vyhovuje všem obvodům. Maximální proudová zatížitelnost je 1A, ztrátový výkon Po < 15W. Celé zařízení lze tedy napájet 9V baterií, nebo DC adaptérem do 30V a min. výstupním proudem 100mA. Stabilizátor je opatřen chladičem, který je nutný při použití síťového adaptéru s napětím nad 12V.

Vstup napájecího obvodu je opatřen filtračním elektrolytickým kondenzátorem C3 pro eliminaci indukčnosti přívodu od napájecího zdroje a přechodových odporů napájecího konektoru a skupinovým blokovacím kondenzátorem C4. Výstup stabilizátoru je veden přes momentové tlačítko, které slouží ke krátkodobému odpojení napájení pro funkci power-on reset mikroprocesoru. Mikroprocesor a CAN budič jsou opatřeny lokálními blokovacími kondenzátory. Připojení napájecího napětí signalizuje červená dioda POWER LED s odběrem 2mA připojená přes odpor R1. Schéma napájecího obvodu je na obr. 2.9.



Obr. 2.9: Schéma napájecího obvodu

2.3.2 Taktování mikroprocesoru

Důležitým krokem je volba frekvence oscilátoru mikroprocesoru. Zařízení využívá pro komunikaci sériový kanál, jehož rychlost je přímo odvozena od frekvence oscilátoru a také CAN komunikaci, jejíž rychlost je taktéž závislá na taktovací frekvenci. Dosažitelné přenosové rychlosti pro různé frekvence jsou na obr. 2.10, str.26.

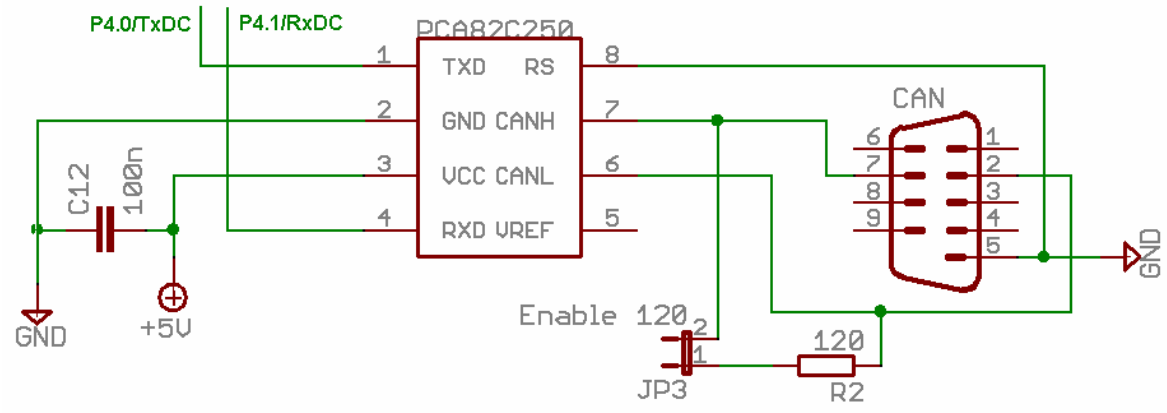
CAN					
Frekvence / přenosová rychlost	8	11,059	12	16	20
20k	OK	OK	OK	OK	OK
100k	OK	OK	OK	OK	OK
125k	OK	OK	OK	OK	OK
250k	OK	OK	OK	OK	OK
500k	OK	OK	OK	OK	OK
1000k	-	-	-	OK	OK
UART					
2400	OK	OK	OK	OK	OK
4800	OK	OK	OK	OK	OK
9600	OK	OK	OK	OK	OK
19200	OK	OK	OK	OK	OK
38400	-	OK	OK	OK	OK
57600	-	OK	-	OK	OK
115200	-	OK	-	-	-

Obr. 2.10: Dosažitelné přenosové rychlosti

K taktování mikroprocesoru jsem zvolil externí krystalový oscilátor na frekvenci 16MHz, což je nejmenší, při které dokáže pracovat CAN při rychlosti 1MB/s. Vyšší frekvence už nežádoucím způsobem zvyšuje spotřebu celého zařízení.

2.3.3 CAN rozhraní

Výstup CAN budiče na sběrnici je vyveden standardním konektorem D-SUB9M se zapojením pinů dle normy ISO 11898, tj. CAN_L na pin č.2, CAN_H na pin č.8 a GND na pin č.5. Na výstupu je připojen jumper JP3, pomocí kterého je možné zapojit ukončovací odpor 120 Ω mezi CAN_H a CAN_L. Schéma zapojení je na obr. 2.11.

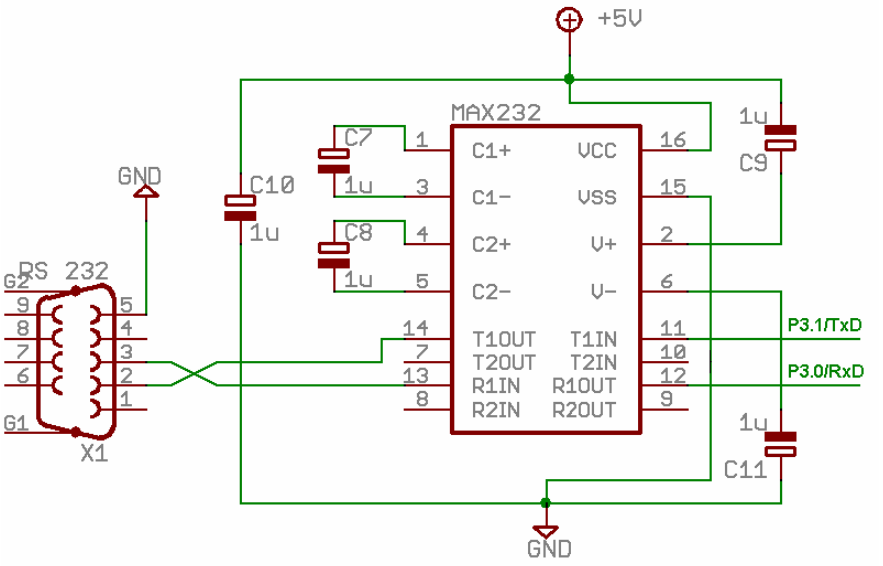


Obr. 2.11: Schéma zapojení CAN výstupu

2.3.4 RS232 rozhraní

Pro komunikaci procesoru s PC přes RS232C jsme zvolil třívodičové zapojení pomocí signálů TxD, RxD a GND. Toto rozhraní podporuje i další řídící signály, ale ze strany procesoru podporovány nejsou. RS232C definuje napěťové úrovně pro log.1 napětí -3 až -15V a pro log.0 napětí +3 až +15V. Výstupy procesoru dosahují úrovně TTL, proto je nutné tyto úrovně upravit pomocí specializovaného obvodu.

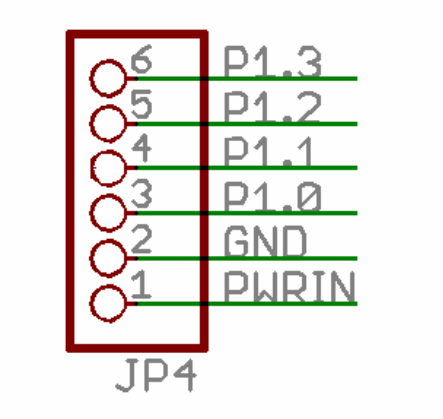
Zvolil jsem obvod MAX232 firmy MAXIM, což je obousměrný konvertor RS232C->TTL napájený ze zdroje +5V a obsahuje dva kanály s inverzí signálu. K dosažení požadovaných úrovní pro RS232C využívá násobiče napětí pomocí 5 kondenzátorů připojených na vstupy obvodu, jejichž hodnota je dle katalogu 1µF. Proudový odběr je 10mA. Výstup je vyveden standardním konektorem D-SUB9F. Signál TxD je připojen na pin č.2, RxD na pin č.3, čili k připojení zařízení k PC je potřeba nekřížený (prodlužovaní) kabel s opačnými konektory na koncích.



Obr. 2.12: Schéma zapojení RS232 výstupu

2.3.5 Vstupní konektor, A/D převodník

Pro připojení snímačů s analogovým výstupem slouží konektor JP4. Je zde vyvedeno napájecí napětí +5V a GND a 4 piny brány P1 (P1.0 až P1.3), první 3 z nich slouží primárně jako vstup A/D převodníku. Je možné je použít i jako vstup/výstup mikroprocesoru používaný pro připojení jiných měřících členů. Schéma zapojení tohoto rozhraní je na obr. 2.13, str. 28.



Obr. 2.13: Zapojení konektoru JP4

Při použití V/V brány jako napěťového vstupu je nutné připojit referenční napětí A/D převodníku v rozsahu 2,4 až 3V na vstup VAREF mikroprocesoru a uzemnit vstup VAGND. Na modulu jsem použil zapojení s obvodem LM336 v pouzdru TO-92, které dává výstupní napětí 2,49V s tolerancí ±0,05V, čili je možné měřit napětí v rozsahu 0 až 2,5V. Výstupem A/D převodníku je při 10-bitovém převodu hodnota 3FFh (plný rozsah) při vstupním napětí úměrnému VAREF a 0h při vstupním napětí úměrnému VAGND. Jakákoliv jiná hodnota mezi VAREF a VAGND je převedena lineárně mezi 0h a 3FFh. Jestliže vstupní napětí je menší než VAGND, resp. větší než VAREF, je výstupem 00h, resp. 3FFh.

Převodník je možné použít pro 8 nezávislých vstupů a může být používán ve dvou režimech. V režimu standardní konverze je převod prováděn jako 8-bitový, v režimu precizní konverze pak jako 10-bitový. V precizním režimu je procesor v tzv. pseudo-idle modu z důvodu snížení šumu okolních obvodů a je nutné použít přerušení k opětovnému „probuzení“ procesoru.

2.4 Výsledná deska plošných spojů

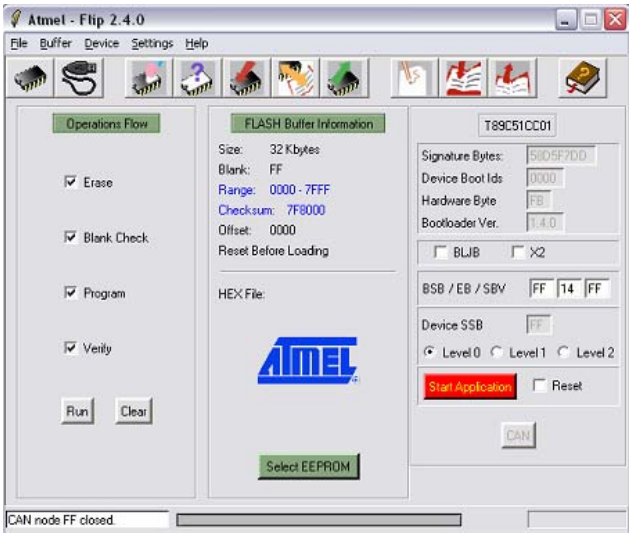
Návrh zapojení elektroniky a desky plošných spojů jsem provedl v programu Eagle 4.11 a pro nestandardní součástky jsem vytvořil vlastní knihovny. Desku plošných spojů jsem navrhl jako jednostrannou ve 4. třídě přesnosti s rozměry 77 x 67 mm. Celkové schéma zapojení, vodivý obrazec desky plošných spojů a osazovací výkres se stručným popisem jsou v přílohách č. 1, 2 a 3.

3 Realizace software

V této kapitole je popsán způsob programování mikroprocesoru, popis programu mikroprocesoru a programu pro PC užívaného ke konfiguraci výsledného CAN uzlu. Dále jsou zde popsány protokoly komunikace mezi CAN uzlem a PC a CAN uzlem a senzorem SHT11.

3.1 Programování mikroprocesoru

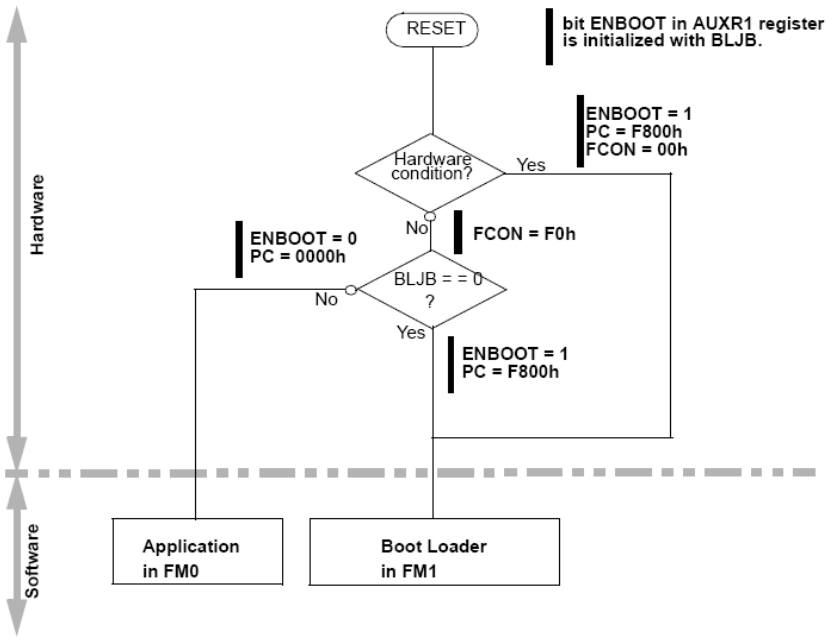
Jak jsem již nastínil v předchozích kapitolách, programování mikroprocesoru jsem zvolil in-system (ISP). Při tomto způsobu programování není nutné procesor vyjímat zapojení a přenášet ho do programátoru. Programování probíhá pomocí sériového rozhraní UART, nebo CAN rozhraní a pomocí obslužného programu, tzv. bootloaderu, což je program ve vyhrazené paměti mikroprocesoru naprogramován od výrobce. Pro tento mikroprocesor existují dvě verze bootloaderu – UART a CAN, přičemž zvolen byl bootloader pomocí UART sériového rozhraní. Bootloader zajišťuje komunikaci se softwarem pro programování a nahrání programu do interní FLASH paměti. Použil jsem programovací software ATLMEL FLIP, který je ke stažení na stránkách výrobce mikroprocesoru. Po spuštění stačí zvolit sériový kanál a přenosovou rychlost. Pro programování jsem používal maximální rychlost 57600 baud/s, kterou je CAN uzel s použitým krystalovým oscilátorem schopen.



Obr. 3.1: Program ATMEL Flip

Software ATMEL FLIP umožňuje kromě zápisu a čtení paměti programu FLASH také čtení a editaci interní paměti EEPROM. Nastavit lze mimo jiné bit BLJB (Boot Loader Jump Bit), což je softwarová podmínka pro načtení bootloaderu.

Obecně bootloader probíhá po resetu mikroprocesoru, pokud jsou splněny následující podmínky: je nastaven bit BLJB, nebo je uzemněn signál PSEN mikroprocesoru. Bit BLJB je standardně nastaven od výroby, takže procesor automaticky spouští bootloader, proto je nutné po prvním naprogramování tento bit pomocí FLIPu vynulovat a k dalšímu programování přecházet přes uzemnění signálu PSEN. Bootovací proces je na obr. 3.2.



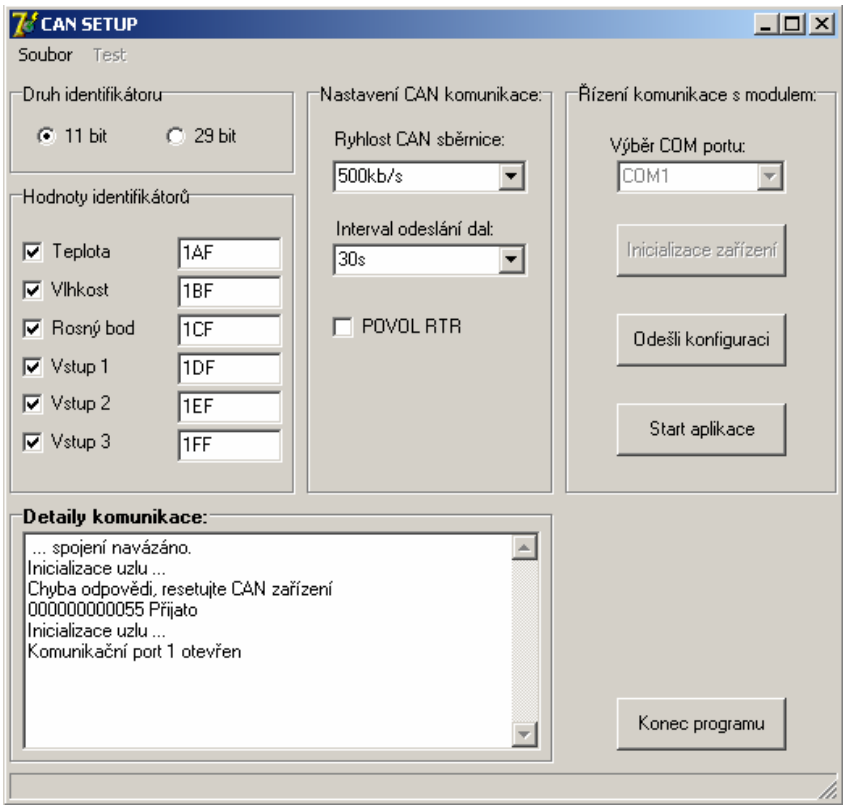
Obr. 3.2: Bootovací proces procesoru [9]

3.2 Konfigurační program pro PC

Program byl navržen a naprogramován v prostředí DELPHI 7. Aplikace se jmenuje CAN SETUP a umožňuje měnit podstatné vlastnosti CAN uzlu z hlediska jeho chování na CAN sběrnici. Jsou to například přenosová rychlost, interval automatického odesílání naměřených dat včetně volby dat, která se mají odesílat na CAN sběrnici. Je možné zvolit identifikátor jednotlivých zpráv včetně formátu CAN 2.0A, či CAN 2.0B. Komunikace s CAN uzlem probíhá pro sériovém rozhraní, proto je nutné, aby počítač, na kterém je CAN SETUP spuštěn, disponoval alespoň jedním volným COM portem.

3.2.1 Hlavní prvky programu

Program je realizován na jednom formuláři a jeho plocha je rozdělena na pět polí, ve kterých jsou sdružovány prvky, která spolu funkčně souvisí. První pole je *Druh identifikátoru*, ve kterém je možné zvolit, zda se naměřené hodnoty budou odesílat na sběrnici CAN se standardním 11 bitovým identifikátorem, či s rozšířeným 29 bitovým. Druhé pole, které s prvním úzce souvisí je volba *Hodnoty identifikátorů*. Zde lze navolit v hexadecimálním vyjádření identifikátor pro každou možnou měřenou veličinu, včetně 3 vstupů A/D převodníku, celkem tedy 6 identifikátorů zpráv. Po spuštění programu je nastaven 11 bitový identifikátor a pole jsou vyplněna s hodnotou 7FF, což je maximální hodnota identifikátoru. Pokud je v poli *Druh identifikátoru* zaškrtnuta volba 29 bitového identifikátoru, pole se vyplní hodnotou maximálního 29 bitového identifikátoru.



Obr. 3.3: Okno aplikace CAN SETUP

Důležitou součástí tohoto pole jsou zaškrtávací políčka (checkbox) u jednotlivých veličin. Slouží k výběru veličiny, která se má na CAN sběrnici vysílat. Pokud je veličina vybrána, je umožněna změna identifikátoru a po odeslání je CAN uzlu oznámeno, že hodnota této veličiny bude odesílána. Pokud pole u veličiny zaškrtnuto není, identifikátor měnit nelze a hodnota veličiny nebude na CAN odesílána.

Třetí pole se jmenuje *Nastavení CAN komunikace* a obsahuje nastavení rychlosti sběrnice CAN volitelnou od 50kb/s do 1Mb/s, interval odesílání dat v rozsahu 5s až 60s a pole POVOL RTR. Pokud je toto pole zaškrtnuto, umožní CAN uzel přijímat dotazovací rámce na data od ostatních uzlů. Identifikátory objektů přijímající žádosti o data pro jednotlivé veličiny jsou o jedničku větší, než hodnoty navolené odesílaným veličinám. Podrobněji budou tyto služby vysvětleny v části věnované řídícímu programu mikroprocesoru, viz kapitola 3.5.

Předposledním polem je okno nazvané *Detaily komunikace*. Sem jsou vypisovány směrem odspoda nahoru informace o úspěšnosti akcí uživatele a informace týkající se přenosu.

Posledním polem je *Řízení komunikace se modulem*. Zde jsou položky používané při komunikaci, jako např. „Inicializace zařízení“, „Odeslání konfigurace“, nebo „Start aplikace“. Před zahájením komunikace je ale nejprve nutné zvolit komunikační port COM. Pokud je zvolený port nepřístupný, nebo neexistuje, je hlášena chyba „Komunikační port COM nelze otevřít“. V opačném případě je do okna zpráv hlášeno úspěšné otevření COM portu. Více o položkách v tomto poli je v následující kapitole.

3.2.2 Odesílání konfigurace

Před odesláním konfigurace je nutné uzel inicializovat stiskem tlačítka „Inicializace uzlu“. Do CAN uzlu je vyslán příkaz k inicializaci (tj. přerušení hlavního programu a přepnutí do režimu čekání na konfigurační data) a pokud je uzel připraven, odešle tuto informaci a aplikace CAN SETUP umožní odeslání konfiguračních dat. Úspěšná i neúspěšná inicializace je hlášena v okně zpráv. Více o chybových hlášeních v kapitole 3.2.4.

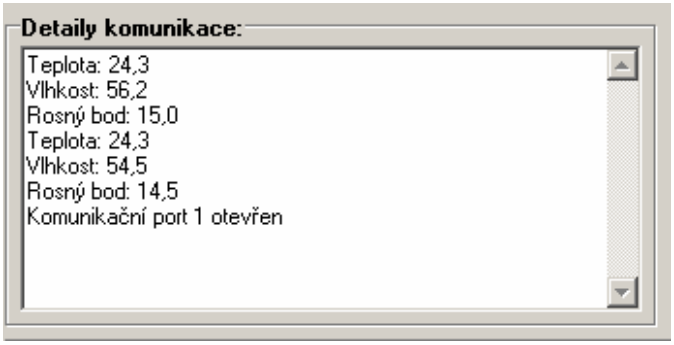
Po stisku tlačítka „Odešli konfiguraci“ jsou procházena jednotlivá pole a v případě identifikátorů jsou kontrolována na správnost zadání. V polích se nesmí vyskytovat nepovolené znaky (malá písmena šestnáctkové soustavy jsou povolena) a identifikátor nesmí mít hodnotu větší, než je rozsah zvoleného typu identifikátoru. Dále jsou kontrolována pole *Rychlost sběrnice* a *Interval odesílání dat*. Tyto nesmí zůstat nevyplněná. Pokud jsou všechna pole správně vyplněna, jsou sestaveny datové packety,

které jsou odeslány. Odeslání každého packetu je doprovázeno výpisem úspěšnosti přenosu do pole *Detaily komunikace*, včetně šestnáctkové hodnoty packetu. Příklad výpisu komunikace je v příloze č. 4. Pokud dorazí všechna data v pořádku, je signalizováno úspěšné dokončení konfigurace, jinak je signalizována chyba přenosu. Pro spuštění uzlu s novou konfigurací stačí kliknout na tlačítko „Start aplikace“, nebo uzel restartovat. Postup popsaný v této kapitole lze opakovat. Pokud ale byla aplikace spuštěna pomoci tlačítka „Start aplikace“, je před další konfigurací nutné uzel znovu inicializovat, což je v programu ošetřeno pomocí povolení, nebo zakázání určitých tlačítek.

Více informací o tvorbě a podobě packetů a zabezpečení přenosu mezi PC a CAN uzlem je v kapitole 3.3.

3.2.3 Testovací měření

Program CAN SETUP umožňuje ověření uloženého nastavení a ověření správné funkce čidla SHT11. V hlavním menu programu je po úspěšném otevření komunikačního portu COM, nebo po stisknutí tlačítka „Start Aplikace“ zpřístupněna položka „Test“. Po kliknutí na „Testovací měření“ v této položce menu je CAN uzel přepnut do režimu, kdy přesměruje data odesílaná na CAN sběrnici ve stejné podobě na sériový port. To znamená, že jsou odesílány pouze hodnoty, které byly při konfiguraci vybrány a v takovém intervalu, který odpovídá nastavené hodnotě. Tyto hodnoty jsou cyklicky vypisovány do okna *Detaily komunikace*. Příklad výpisu testovacího měření je na obr. 3.4. Pro ukončení testovacího měření stačí kliknout na tlačítko „Start aplikace“. Inicializací CAN uzlu je testovací měření také ukončeno.



Obr. 3.4: Detail výpisu testovacího měření

3.2.4 Chybová hlášení

Při běhu programu se může uživatel setkat s následujícími chybovými hlášeními:

- Chyba odpovědi, resetujte CAN zařízení – byla obdržena špatná odpověď na příkaz Inicializace uzlu, Start aplikace, nebo Kontrolní měření.
- … chyba spojení – v časovém limitu nebyla obdržena odpověď. Je vhodné resetovat CAN uzel.
- Chyba přenosu, opakuji. – data nebyla přenesena správně, vysílání je opakováno
- Chybný přenos, resetujte CAN zařízení – chyba přenosu 10x po sobě, vysílání je ukončeno. Je vhodné restartovat také program CAN SETUP.
- Příliš velký identifikátor – hexadecimální hodnota identifikátoru je větší, než rozsah zvoleného druhu identifikátoru
- Špatně zadaný identifikátor – identifikátor obsahuje nepovolené znaky
- Zadejte rychlost CAN sběrnice! – není zadána rychlost sběrnice
- Zadejte interval posílání dat! – není zadán interval posílání dat
- Komunikační port COM nelze otevřít – komunikační port je obsazen, nebo neexistuje

3.3 Protokol komunikace mezi CAN uzlem a PC

Komunikace probíhá po sériové lince RC232C. Přenosová rychlost je nastavena na 38 400 Baud/s a přenos probíhá poloduplexně, kdy vysílá pouze jedna strana a druhá přijímá vysílané znaky. V jednom rámci RC232C je přeneseno vždy 8 bitů uvozený jedním start bitem a jedním stop bitem. RS232C umožňuje připojit paritní bit, ale v tomto případě není využíván. Komunikace probíhá na principu vysílání příkazů v podobě řídících znaků nebo uživatelských dat v podobě packetů z PC, přičemž od CAN uzlu je očekávána odpověď.

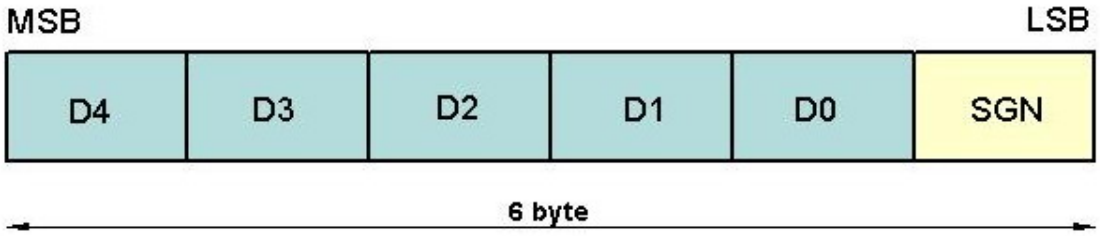
V případě příkazů uzel odpovídá předem definovanou odpovědí, která značí, že CAN uzel příkaz ve správné podobě přijal a začal vykonávat požadovanou akci. Všechny řídící znaky i odpovědi jsou jednobajtové a představují akce vyvolané uživatelem, nebo událostí během komunikace. Jejich přehled je uveden na obr. 3.5, str. 35.

PŘÍKAZ	ŘÍDÍCI ZNAK (HEX)	ODPOVĚĎ (HEX)
Inicializace uzlu	55	66
Start aplikace	77	88
Testovací měření	99	bez odpovědi
Data OK	AD	bez odpovědi
Data NOT OK	DA	bez odpovědi

Obr. 3.5: Přehled řídících příkazů a odpovědí

3.3.1 Struktura odesílaných dat a zabezpečení přenosu proti chybám

Jak již bylo řečeno, uživatelská data jsou přenášena v podobě packetů. Tyto packety mají délku 6 bajtů a jak bude ukázáno dále, tato délka je vhodná pro přenesení konfiguračních dat právě jedné měřené veličiny a taktéž lze jedním packetem přenést nastavení CAN komunikace. Struktura packetu je na obr. 3.6.



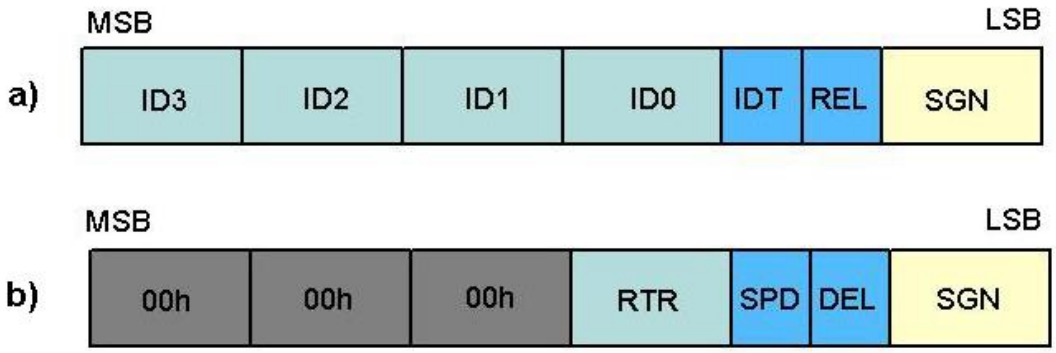
Obr. 3.6: Struktura datového packetu

Aplikace CAN SETUP umožňuje nastavit konfiguraci šesti měřených veličin a spolu s údaji o nastavení CAN sběrnice stačí na přenesení kompletní konfigurace CAN uzlu 7 packetů (celkem 42 bajtů). Tyto packety jsou rozlišeny pomocí prvního bajtu na místě nejméně významného bajtu (LSB), který je označen SGN. Tento „značkovací“ bajt nese informaci o tom, jaká data jsou přenášena a CAN uzel je podle této informace ukládá do paměti EEPROM. Zbylých pět nejvyšších bajtů D0 až D5 již nese informace nastavené uživatelem.

Podoba packetu při zakódování konfigurace měřených veličin je na obr. 3.7a, str. 36. Po bajtu SGN následuje bajt rozdělený na dvě části. Spodní čtyři bity jsou označené REL (relevancy) a odpovídají checkboxům u jednotlivých veličin v aplikaci CAN SETUP, kterými jsou vybrána data, jež se mají posílat na CAN sběrnici. Pokud je

chceckbox u příslušné veličiny zaškrtnut, hodnota REL odpovídá hexadecimální hodnotě 0Bh.V opačném případě 0Eh.

Druhá polovina bajtu je doplněna 4 bity IDT (identifer type) který značí, jestli se jedná o 11 bitový (hodnota 0Ah), nebo 29 bitový (hodnota 0Fh) identifikátor. Následují 4 bajty ID0 až ID3, do kterých je uložen identifikátor měřené veličiny postupně od nejnižšího bajtu (ID0). Pokud je přenášen pouze 11 bitový identifikátor, jsou nejvyšší dva bajty doplněny nulami.



Obr. 3.7: Podoby packetů při komunikaci mezi PC a CAN uzlem

Podoba packetu při zakódování nastavení CAN komunikace je na obr. 3.7b. Opět je přítomen bajt SGN po kterém následují čtyři bity DEL (delay), ve kterých je zakódován výběr prodlevy odesílání dat. Hexadecimální hodnoty 0Ah, 0Bh, 0Ch a 0Dh odpovídají postupně zvoleným hodnotám prodlevy 5 až 60s.

V dalších čtyřech bitech SPD je zakódována zvolená rychlost komunikace. Hexadecimální hodnoty 0Ah, 0Bh, 0Ch a 0Dh odpovídají postupně zvolené přenosové rychlosti 50kB/s až 1MB/s

Bajt RTR představuje hodnotu 0FFh v případě, že je povoleno přijímat dotazovací rámce (zaškrtnuto políčko „Povolit RTR“ v aplikaci CAN SETUP). V opačném případě RTR představuje hodnotu 0AAh.

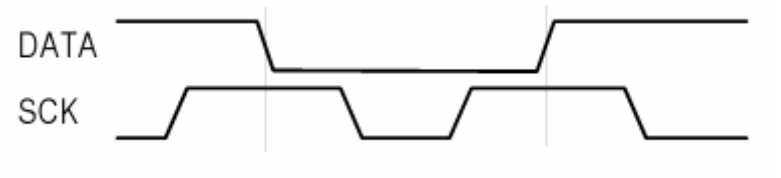
V případě odesílání uživatelských dat je potřeba zaručit konzistenci přenášené informace. Toto je zabezpečeno tím, že CAN uzel pošle každý packet zpět do PC ve stejné podobě, v jaké ho obdržel. V PC je packet z CAN uzlu porovnán se podobou packetu který byl původně odvysílán a pokud se shodují, je umožněno CAN uzlu tento packet použít a zapsat do paměti EEPROM vysláním řídícího znaku *Potvrzení platnosti*

dat. V opačném případě je vyslán řídící znak *Zamítnutí platnosti dat*, CAN uzel poškozený packet vymaže a čeká na opakování přenosu. Pokud během komunikace dojde k 10 po sobě jdoucím chybám, komunikace je stornována.

3.4 Komunikace se senzorem SHT11

Senzor SHT11 je připojen pomocí 2-vodičové obousměrné sběrnici k pinům P1.4 (DATA) a P1.5 (SCK) procesoru. Signál SCK je používán k synchronizaci komunikace a signál DATA je používán k přenosu dat. Přenos dat je zabezpečen pomocí polynomiáního součtu CRC-8, přičemž aplikační zpráva výrobce obsahuje 2 metody výpočtu – bitovou a bajtovou. Bitová spočívá v online výpočtu pomocí bitových operací, bajtová využívá 256 bajtů dlouhé vyhledávací (look-up) tabulky.

Komunikaci zahajuje tzv. startovací sekvence, která se skládá z nastavení DATA do log.0 zatímco SCK je v log.1, následujícího pulsu SCK signálu a opětovného nastavení DATA do log.1 zatímco SCK je v log.1, viz obr. 3.8.



Obr. 3.8: Startovací sekvence [12]

Po startovací sekvenci následuje příkaz, který se skládá ze tří adresních bitů (je vždy roven „000“) a pěti bitů příkazu, dle obr. 3.9, str. 38. Jestliže je příkaz přijat, SHT11 odpoví ACK signálem, tj. nastavením DATA do log.0 se závěrnou hranou osmého SCK signálu. Se závěrnou hranou devátého SCK signálu je signál DATA „uvolněn“ zpět do log.1. Po přijetí ACK sekvence musí procesor počkat na dokončení měření. Tato doba je přibližně 11/55/210ms pro 8/12/14 bitové měření. Ukončení měření signalizuje SHT11 nastavením DATA do log.0 a následuje přenos 2 bajtů měřených dat (první MSB) a jednoho bajtu CRC součtu, přičemž procesor musí odpovědět na každý bajt ACK signálem. Komunikace je ukončena vysláním ACK signálu po přenosu CRC součtu. Pokud není využívána kontrola pomocí CRC, může

procesor ukončit komunikaci ihned po přenosu naměřených dat „podržením“ DATA v log.1 při odpovědi na druhý bajt.

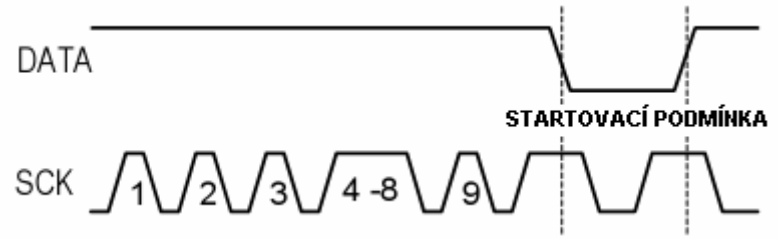
PŘÍKAZ	KÓD PŘÍKAZU
Měření teploty	00011
Měření vlhkosti	00101
Čtení status registru	00111
Zápis do status registru	00110
Soft reset	11110

Obr. 3.9: Řídící příkazy SHT11

Příkazem Soft reset se restartuje senzor, registry se nastaví na defaultní hodnoty a je nutná počkat 11ms před dalším příkazem.

3.4.1 Reset komunikace

Pokud je spojení nějakým způsobem přerušeno, sekvencí dle obr. 3.10 je komunikace resetována (nikoliv status registr). Sekvence sestává s devíti a více SCK pulzů při ponechání DATA v log.1. a musí následovat startovací sekvence před dalším zápisem příkazu.



Obr. 3.10: Sekvence resetující komunikaci [12]

3.5 Řídící program mikroprocesoru

Řídící program je uložen v programovatelné paměti FLASH mikroprocesoru a je napsán v jazyce C v prostředí KEIL µVision. Program zajišťuje komunikaci s čidlem SHT11, které slouží jako senzor teploty a vlhkosti a z těchto hodnot je počítán rosný bod. Dále je cyklicky převáděno napětí na vstupech I/O brány do digitální podoby pomocí A/D převodníku a tyto hodnoty jsou spolu s hodnotami získanými pomocí čidla SHT11 posílána na CAN sběrnici, přičemž každé hodnotě je přiřazen jeden objekt zpráv CAN řadiče s vlastním identifikátorem (celkem 6 objektů). Další 3 objekty zpráv jsou konfigurovány pro příjem požadavků na data (teploty, vlhkosti a rosného bodu) s automatickou odpovědí. Dále program komunikuje s nadřazeným systémem prostřednictví sériového kanálu.

3.5.1 Hlavní linie programu

Hlavní program běží v nekonečné smyčce a je řízen pomocí čtyř druhů přerušení: čítače/časovače 1, sériového kanálu, A/D převodníku a CAN řadiče. Po spuštění programu je proveden reset CAN řadiče a jsou vynulovány registry jednotlivých objektů zpráv. Poté jsou z paměti EEPROM načteny uložené konfigurační parametry, jsou nastaveny identifikátory jednotlivých objektů zpráv, přenosová rychlost CAN sběrnice a časování bitů. Dále jsou nastaveny čítače/časovače, A/D převodník a jsou upraveny priority přerušení.

Hlavní cyklus programu je vykonáván při přerušení od přetečení čítače/časovače1 (č/č 1), který pracuje v módu 16 bitového časovače s frekvencí přetečení 20Hz. Pomocí tohoto přerušení je generována časová prodleva mezi jednotlivými vysíláními naměřených hodnot na CAN sběrnici.

Pokud je dosaženo požadované prodlevy (počtu přerušení časovače), jsou vyžádány hodnoty od čidla SHT11 a jsou upraveny podle vztahů 2.1 až 2.4 popsaných v kapitole 2.2.3. Dále jsou převedeny hodnoty napětí ze zvolených vstupů A/D převodníku. Poté, co jsou všechny hodnoty k dispozici, jsou uloženy do jednotlivých objektů zpráv a postupně vysílány na CAN sběrnici. Tímto se uzavírá smyčka hlavního

programu a celý hlavní cyklus se opakuje. Popis jednotlivých operací vykonávaných během hlavního cyklu je v následujících kapitolách.

Nastavení č/č 1 se provádí pomocí registru TMOD a to horních čtyřech bitů 4-7, spodní čtyři bity se týkají č/č 0. Význam jednotlivých bitů je na obr. 3.11.

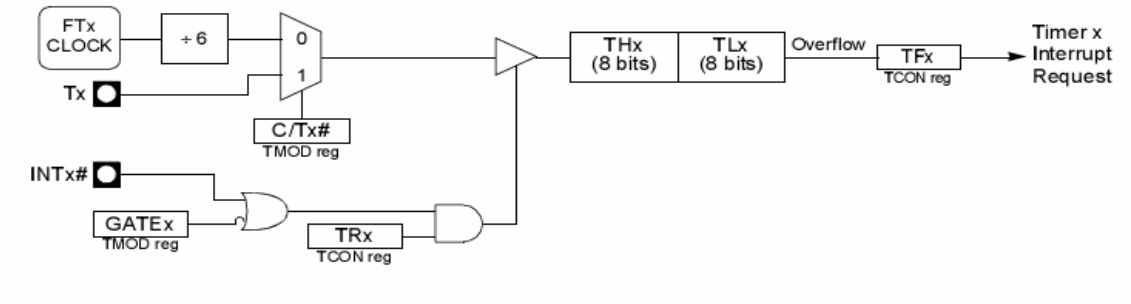
Spuštění/zastavení č/č 1 probíhá pomocí bitu TR1 v registru TCON, při přetečení je nastaven příznak TF1 v témže registru. Povolení přerušení od č/č 1 se realizuje nastavením bitu ET1 v registru IEN0. K těmto třem bitům je možné přistupovat přímo (bitově).

7	6	5	4	3	2	1	0
GATE1	C/T1	M11	M01	GATE0	C/T0	M10	M00

Obr. 3.11: Registr TMOD

- GATE1 – řízení „hradlování“. Při GATE1 = 1 je č/č 1 spuštěn pouze pokud je vstup INT1 = 1 a TR1 = 1. Při GATE1 = 0 je č/č 1 spuštěn při TR1 = 1.
- C/T1 – volba režimu čítač/časovač. Pokud C/T1 = 0, jde režim časovače. Pokud C/T1 = 1, jde o režim čítače vnějších událostí na pinu T1.
- M11, M01 – volba pracovního módu. Byl zvolen mód 1 (M11 = 0, M01 = 1), kdy je č/č 1 tvořen dvěma 8 bitovými registry TH1 a TL1, které dohromady tvoří 16 bitový registr.

Vnitřní zapojení č/č 1 je na obr. 3.12, str. 41. Vstupní frekvence č/č 1 je odvozena od frekvence FTx, která je rovna $\frac{F_{OSC}}{2}$. Dále je frekvence FTx dělena 6, takže výsledná vstupní frekvence č/č 1 je $\frac{16MHz}{12}$. V módu 16 bitového časovače je hodnota přetečení 65536, tudíž při nastavení TH1, TL1 = 0 při každém přetečení je výsledné zpoždění rovno $65536 \cdot \frac{12}{16MHz} = 0{,}0492s \approx 20Hz$. Není to přesná hodnota 20Hz, ale dostačuje pro realizaci zpoždění např. 5 vteřin (100 cyklů přetečení je 4,92s), protože určitou dobu trvá, než proběhnou všechna měření a výpočty.



Obr. 3.12: Vnitřní zapojení čítače/časovače 1 [9]

Nastavení č/č 1 je v programu realizováno příkazy ve funkci *T1_init* :

```
TMOD &= 0x0F;
TMOD |= 0x10;
TH1 = 0x00;
TL1 = 0x00;
ET1 = 1;
```

3.5.2 Měření pomocí čidla SHT11

Komunikace s čidlem a vlastní měření probíhá v rámci hlavní linie programu při potřebě vyslat aktuální data na CAN sběrnici (příp. do PC). Pro komunikaci s čidlem SHT11 jsou vytvořeny následující funkce:

- *char s_write_byte(unsigned char value)* – zapíše 1 bajt příkazu na sběrnici
- *char s_read_byte(unsigned char ack)* – přečte 1 bajt ze sběrnice
- *void s_transstart(void)* – zápis startovací sekvence na sběrnici
- *void s_connectionreset(void)* – reset komunikace s čidlem
- *char s_measure(unsigned char *p_value, unsigned char *p_checksum, unsigned char mode)* – funkce zajišťující vlastní měření hodnot
- *void calc_sth11(float *p_humidity ,float *p_temperature)* – funkce provádí převod digitálních hodnot z čidla SHT11 na fyzikální veličiny (linearizaci vlhkosti, kompenzaci vlhkosti vlivem teploty a převod teploty)
- *float calc_dewpoint(float h,float t)* – vypočte hodnotu rosného bodu
- *void run_measure (void)* – funkce provede kompletní měření včetně všech potřebných výpočtů pomocí volání výše uvedených funkcí. Mimo jiné zajišťuje i měření napětí pomocí A/D převodníku

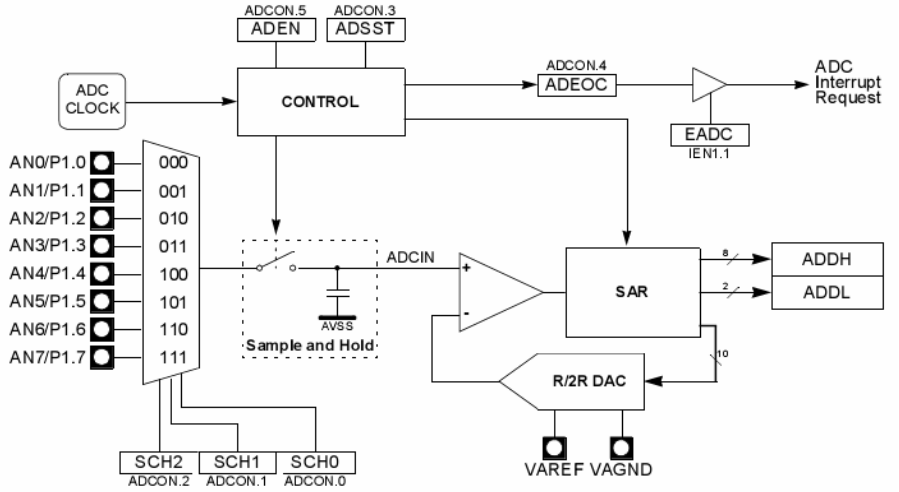
Cyklus měření probíhá následovně. Nejprve je na sběrnici vyslána startovací sekvence, po té následuje vyslání příkazu „Měření teploty“ (viz obr. 3.9, str. 38) a po dokončení měření je ze sběrnice přečtena dvoubajtová hodnota včetně kontrolního součtu CRC-8 (i když se s ním dále nepracuje). Stejným způsobem je získána hodnota vlhkosti. Proměnné teploty a vlhkosti jsou definovány pomocí struktury union, která umožňuje dvojí podobu proměnné, protože jsou přijímány jako celočíselný datový typ *integer* a pro další zpracování je třeba typ *float*.

```
typedef union  {
unsigned int i;
float f;
} value;
```

Po přetypování na datový typ *float* je vypočtena hodnota teploty a provedena linearizace vlhkosti a její teplotní kompenzace. Z těchto hodnot je vypočítán rosný bod a takto získané hodnoty jsou již připraveny pro vyslání na CAN sběrnici.

3.5.3 Měření pomocí A/D převodníku

Stejně jako měření pomocí čidla SHT11 je měření napětí pomocí A/D převodníku prováděno ve funkci *void run_measure (void)*. Jsou zde postupně procházeny kanály 0-2 A/D převodníku, které odpovídají vstupním pinům 3-5 konektoru JP4, pro něž je spouštěn převod. Ukončení převodu je signalizováno přerušením, při jehož obsloužení jsou uloženy hodnoty napětí zapisovány do pole pro pozdější použití. A/D převodník je nastaven jako 8 bitový se vzorkovací frekvencí 125kHz. Vnitřní zapojení A/D převodníku je na obr. 3.13.



Obr. 3.13: Vnitřní zapojení A/D převodníku [9]

Převodník je nastaven a řízen pomocí třech registrů. Registr ADCF udává, který ze vstupů 0-7 A/D převodníku bude moci být využit pro převod. Registr je osmibitový a přítomnost jedničky v jednotlivých bitech 0-7 značí, že odpovídající vstup bude použit. Frekvence vzorkování je nastavena pomocí registru ADCLK. Hodnota prvních pěti bitů je tzv. prescaler, který udává, kolikrát bude frekvence hodin procesoru vydělena. Výsledná frekvence je použita jako vzorkovací. Povolení přerušení od A/D převodníku se realizuje nastavením bitu EADC v registru IEN1 (bitově adresovatelný). Výsledek převodu je v registrech ADDH a ADDL (v případě 8 bitového převodu je výsledek pouze v registru ADDH).

Řízení převodníku je možné pomocí registru ADCON. Význam jednotlivých bitů je na obr. 3.14.

7	6	5	4	3	2	1	0
-	PSIDLE	ADEN	ADEOC	ADSST	SCH2	SCH1	SCH0

Obr. 3.14: Registr ADCON

- PSIDLE - jestliže je nastaven, udává, že bude probíhat 10 bitový převod v tzv. pseudo-idle módu. Vynulování značí standardní 8 bitový převod
- ADEN – jestliže je nastaven, A/D převodník je povolen. V opačném případě je režimu StandBy, kdy má minimální odběr.
- ADEOC – příznak ukončení převodu. Musí být nulován softwarově.
- ADSST – nastavením se spustí převod. Je nulován hardwarově.
- ACH2:SCH0 – binární kombinací je vybírán kanál 0-7, který se použije při následujícím převodu

Nastavení A/D převodníku je realizováno příkazy ve funkci *ADC_init* :

```
ADCF &= ~0xFF;
ADCF = 0x07;
ADCLK = 0x06;
ADCON = 0x20;
EADC = 1;
```

3.5.4 Obsluha EEPROM

Pro čtení a zápis do paměti EEPROM jsou definovány dvě funkce:

- *unsigned char rd_eeprom_byte (unsigned char xdata *adress)* - čtení
- *void wr_eeprom_byte (unsigned char xdata *address, unsigned int hodnota)* - zápis

K dispozici jsou 2kb paměti EEPROM na adresách 0000h až 07FFh a k jejímu řízení je k dispozici registr EECON, viz obr. 3.15. K mapování této oblasti je použit ukazatel do paměti **address,* který ukazuje do oblasti x*data.*

Zápis do EEPROM se pak skládá z namapování EEPROM pomocí uložení hodnoty 02h do registru EECON, uložením požadované hodnoty na adresu **address* a postupným zapsáním hodnot 50h a A0h do registru EECON.

Čtení z EEPROM spočívá v testování příznaku EEBUSY a jestliže příznak není nastaven, je mapována EEPROM a přečtena hodnota z adresy **address* .

7	6	5	4	3	2	1	0
EEPL3	EEPL2	EEPL1	EEPL0	-	-	EEE	EEBUSY

Obr. 3.15: Registr EECON

- EEPL3:EEPL0 – do těchto čtyřech bitů ze zapisují hodnoty 50h a A0h jako řídící příkazy umožňující zápis do EEPROM
- EEE – nastavením se mapuje EEPROM pro zápis, nebo čtení
- EEBUSY – je nastaven hardwarově, pokud probíhá programování paměti EEPROM

3.5.5 Obsluha CAN rozhraní

CAN rozhraní slouží k vysílání naměřených dat na CAN sběrnici a uskutečňuje se pomocí objektů zpráv CAN řadiče. K dispozici je celkem 15 objektů zpráv, které lze nakonfigurovat jako přijímač, nebo jako vysílač s vlastním identifikátorem. Je využíváno celkem 9 objektů, které jsou nakonfigurovány podle obr. 3.16, str. 45.

INDEX OBJEKTU	TYP OBJEKTU	VELIČINA	IDENTIFIKÁTOR
0	vysílací	teplota	ID_teplota
1	vysílací	vlhkost	ID_vlhkost
2	vysílací	rosný bod	ID_rosný_bod
3	vysílací	vstup 1	ID_vstup_1
4	vysílací	vstup 2	ID_vstup_2
5	vysílací	vstup 3	ID_vstup_3
6	přijímací + RTR	teplota	ID_teplota+1
7	přijímací + RTR	vlhkost	ID_vlhkost+1
8	přijímací + RTR	rosný bod	ID_rosný_bod+1

Obr. 3.16: Konfigurace objektů zpráv

Nastavení objektů zpráv spolu s nastavením přenosové rychlosti a časování bitů probíhá po každém spuštění/resetu CAN uzlu a po každém nakonfigurování pomocí aplikace CAN SETUP pomocí funkcí *CAN_general_init* a *CAN_transmit_init*, přičemž potřebné hodnoty jsou načteny z paměti EEPROM. Vysílacím objektům s indexy 0 – 5 je nastaven 11 bitový, nebo 29 bitový identifikátor dle uživatelského nastavení. Přijímací objekty s indexy 6-8 jsou v případě povolení příjmu požadavků na data (RTR) nakonfigurovány pro příjem požadavků s automatickou odpovědí a s identifikátorem o jedničku vyšším, než jim odpovídající vysílací objekty, viz obr. 3.16.

Vysílání uživatelem zvolených naměřených hodnot probíhá v obsluze přerušení od čítače/časovače 1. Do jednotlivých vysílacích objektů zpráv 0 až 5 jsou postupně uloženy hodnoty odpovídajících veličin, je nastaven počet vysílaných datových bajtů a je spuštěn přenos. Vysílání hodnot probíhá bez nutnosti dalšího řízení.

V případě objektů zpráv 6-8 je nutné je nakonfigurovat a ukládat do jejich datových registrů aktuální hodnoty pro případnou odpověď na požadavek nejen v době ukládání dat do vysílacích objektů 0-5 (v přerušení od č/č 1), ale je nutné zabezpečit jejich rekonfiguraci a naplnění aktuálními daty v případě, že přijde požadavek na data a tyto jsou odvysílána, protože po odvysílání změní objekt zpráv svůj stav a je zakázán. V případě, že by během prodlevy cyklického vysílání přišlo více požadavků na data, byl by obsloužen pouze první z nich. K tomuto účelu je u objektů 6-8 povoleno vyvolání přerušení při odvysílání zprávy, kterým je vyvoláno přerušení od CAN řadiče. V obsluze tohoto přerušení jsou objekty znovu nakonfigurovány jako přijímací s automatickou odpovědí a jsou znovu naplněny naposled změřenými hodnotami.

Přenosová rychlost CAN sběrnice je volitelná pomocí programu CAN SETUP. Pro každou přenosovou rychlost jsou napevno zvoleny hodnoty časování se

vzorkovacím bodem 75%, které jsou shrnuty na obr. 3.17. Význam jednotlivých hodnot je popsán v kapitole 1.4. Hodnoty BRP, PRS, PHS1, PHS2 a SJW se do registrů ukládají vždy o jedničku menší, než je jejich skutečná hodnota.

CAN SPEED [kB/s]	BRP	PRS	PHS1	PHS2	SJW	CELKEM t_q
50	9	3	6	3	1	16
100	4	3	6	3	1	16
500	0	3	6	3	1	16
1000	0	1	2	1	0	8

Obr. 3.17: Nastavení časování CAN sběrnice

K nastavení CAN řadiče je k dispozici 34 konfiguračních registrů. Vzhledem k jejich počtu je popis používaných a nejdůležitějších registrů je uveden v příloze č. 5.

3.5.6 Obsluha sériového kanálu

Sériový kanál (UART) slouží ke konfiguraci CAN uzlu, resp. jeho přenosových vlastností směrem na CAN sběrnici, pomocí programu CAN SETUP pro PC. Detaily přenosu dat jsou popsány v kapitole 3.2. Sériový kanál pracuje v režimu 1 (8 bit UART), přenosová rychlost je odvozena od čítače/časovače 2 pracujícího v režimu generátoru přenosové rychlosti 38 400 baud/s. Pro příjem i vysílání slouží registr SBUF, do kterého jsou ukládána přijatá i vysílaná data. V případě příchozích dat je generováno přerušení od sériového kanálu a v jeho obsluze je příchozích 8 bitů testováno na přítomnost některého z řídících příkazů. V případě, že program je ve stavu cyklického vysílání dat na CAN sběrnici, jsou akceptovány pouze řídící příkazy *Inicializace uzlu* a *Testovací měření*.

Pokud je obdržen příkaz *Inicializace uzlu*, hlavní cyklus programu je přerušen, odešle se definovaná odpověď a spustí se podprogram příjmu konfiguračních dat, který představuje funkce *recept_data*. V opačném případě je znak odeslán zpět a program pokračuje tam, kde byl přerušen.

Jestliže došlo k inicializaci, je očekáván příchozí datový packet o velikosti 6 bajtů, nebo řídící znak *start aplikace*, který ukončí příjem dat a je obnoven hlavní cyklus programu. Spuštění tohoto cyklu náhodným příchozím znakem shodným s řídícím znakem příkazu S*tart aplikace* je zamezeno tak, že příchozí packet obsahuje

na místě prvního bajtu „značkovací“ bajt SGN, který je odlišný od řídících znaků. V případě, že byl bajt SGN přijat, je náhodnému spuštění hlavního cyklu programu zamezeno softwarovým příznakem a je umožněno až po přijetí kompletního packetu.

Jednotlivé příchozí bajty packetu jsou ukládány do vstupního bufferu a v případě, že byl již přijat celý packet, je odeslán zpět a je očekáváno potvrzení konzistence dat. Jestliže se tak stane, relevantní data obsažená v packetu jsou zapsána do paměti EEPROM a je očekáván další packet, nebo příkaz *Start aplikace*. Jestliže je konzistence odmítnuta, vstupní buffer je vynulován a přenos packetu se opakuje.

Po ukončení konfigurace je před obnovením hlavního cyklu programu resetován CAN řadič a je načtena kompletní konfigurace z paměti EEPROM voláním funkcí *CAN_general_init* a *CAN_transmit_init*.

Pokud je během hlavního cyklu programu obdržen příkaz *Testovací měření*, tento cyklus se nepřeruší, pouze jsou data vysílaná na CAN sběrnici přesměrována na sériový port. Zpětné přesměrování toku dat na CAN sběrnici je možné vysláním příkazu *Start aplikace*, nebo hardwarovým resetem.

Sériový kanál má k dispozici vlastní konfigurační registr SCON. Význam jednotlivých bitů registru je na obr. 3.18.

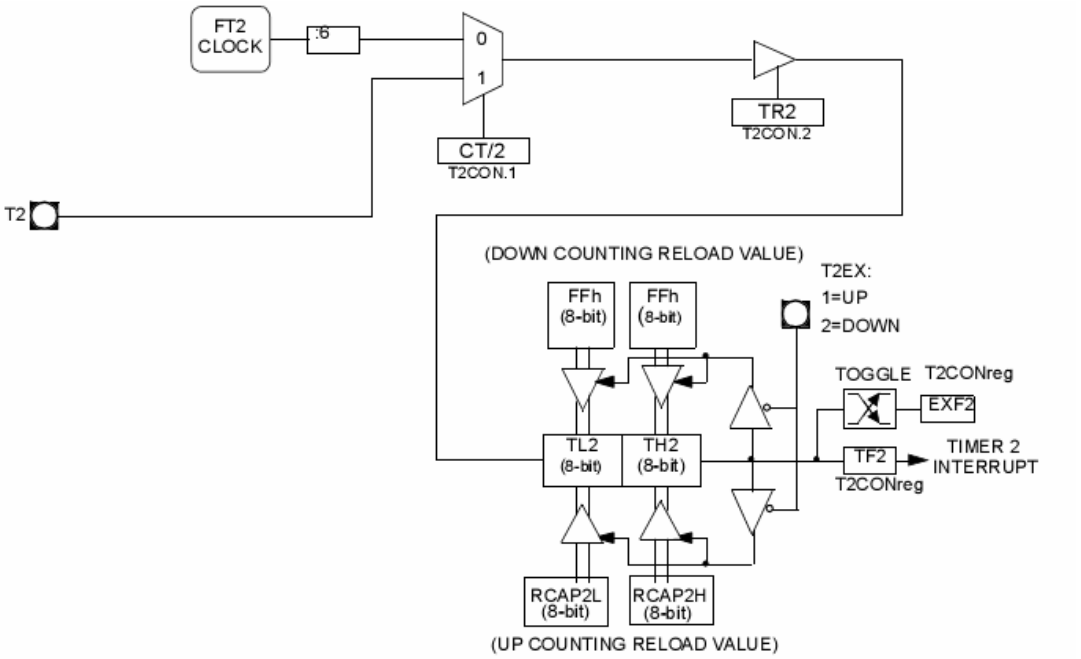
7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

Obr. 3.18: Registr SCON

- SM1:SM0 – volba módu sériového kanálu
  - 0 0 – posuvný registr, rychlost pevná
  - 0 1 – 8 bitový UART, rychlost volitelná
  - 1 0 – 9 bitový UART, rychlost pevná
  - 2 1 – 9 bitový UART, rychlost volitelná
- SM2 – povolení/zakázání multiprocesorové komunikace
- REN – povolení/zakázání příjmu dat
- TB8,RB8 – hodnota devátého bitu při vysílání, resp. příjmu
- TI, RI – příznak odvysílání, resp. příjmu dat

Přerušení od sériového kanálu je povoleno bitem ES v registru přerušení IEN0.

Čítač/časovač 2 je 16 bitový s automatickým načtení předvolby. Podobně jako č/č 1 je tvořen dvěma 8 bitovými registry TH2 a TL2 pouze s tím rozdílem, že je do nich při přetečení automaticky načtena hodnota předvolby z registrů RCAP2H a RCAP2L. Vnitřní zapojení je na obr. 3.19.



Obr. 3.19: Vnitřní zapojení čítače/časovače 2 [9]

Čítač/časovač 2 je pomocí registru T2CON nastaven jako generátor přenosové rychlosti sériového kanálu o hodnotě 38400 baud/s. Význam jednotlivých bitů registru T2CON je na obr. 3.20.

7	6	5	4	3	2	1	0
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2

Obr. 3.20: Registr T2CON

- TF2 – příznak přetečení, musí být nulován softwarově. Není nastavován, pokud je zvolen režim generátoru přenosové rychlosti sériového kanálu.
- EXF2 – vnější příznak. Je nastaven, jestliže zachycení nové hodnoty, nebo přednastavení je způsobeno sestupnou hranou na pinu T2EX, v případě, že je tento pin povolen bitem EXEN2.
- RCLK – jestliže je nastaven, slouží č/č 2 jako generátor rychlosti sériového kanálu při příjmu.

- TCKL – obdobně jako RCLK, pouze jde o rychlost vysílání
- EXEN2 – povolení/zakázání sledování událostí na pinu T2EX
- TR2 – spuštění/zastavení čítače/časovače 2
- C/T2 – volba čítač/časovač

  0 – časovač

  1 – čítač vnějších událostí na pinu T2
- CP/RL2 – volba zachycení nebo přednastavení při sestupné hraně pinu T2EX

16 bitová hodnota předvolby RCAP (složená z registrů RCAP2H a RCAP2L) je vypočtena z následujícího vztahu:

$$RCAP = 65536 - \frac{F_{OSC}}{32 \cdot BaudRate} = 65536 - \frac{16MHz}{32 \cdot 38400} = 65523 = FFF3h \quad (3.1)$$

Nastavení č/č 2 a sérového kanálu je provedeno funkcí *T2_init*:

```
SCON = 0x50;
 T2CON &= 0xF0;
T2CON |= 0x30;
TH2 = 0xFF;
TL2 = 0xF3;
RCAP2H = 0xFF;
RCAP2L = 0xF3;
ES = 1;
```

4 Ověření funkce

4.1 Ověření CAN komunikace

K ověření funkce navrhovaného uzlu směrem do CAN sběrnice byl použit vývojový systém s firmy Hitex Ltd. S procesorem Infineon C167CR-LM. Dispozici je CAN řadič s 15 objekty zpráv a budič sběrnice Philips PCA82C250. Z dostupných prostředků byl použit také 2 řádkový LCD displej k zobrazení příchozích zpráv.

Program pro mikroprocesor C167CR-LM byl napsán v jazyce C v prostředí KEIL µVision. Pro ověření funkce byla zvolena přenosová rychlost 100kB/s se vzorkovacím bodem 75% při použití standardního 11 bitového identifikátoru. Použito bylo celkem 6 objektů zpráv s indexy 0-5, z toho první tři jako přijímací objekty teploty, vlhkosti a rosného bodu, zbylé tři objekty jako vysílací žádostí o data týchž veličin. Nastavení objektů zpráv je na obr. 4.1.

OBJEKT	IDENTIFIKÁTOR	FUNKCE
0	1AA	přijímací
1	1BA	přijímací
2	1CA	přijímací
3	1AB	vysílací RTR
4	1BB	vysílací RTR
5	1CB	vysílací RTR

Obr. 4.1: Nastavení objektů zpráv C167CR-LM

Navržený CAN uzel byl nastaven pomocí programu CAN SETUP pro vysílání teploty, vlhkosti a rosného bodu v časovém intervalu 5s rychlostí 100kB/s s možností přijímat požadavky na data. Identifikátory vysílacích objektů byly nastaveny totožně s přijímacími objekty na straně C167CR-LM.

Příjem dat je realizován v nekonečné smyčce, kdy jsou všechny objekty testovány na příznak nových dat. V případě, že je tento příznak nastaven, jsou příchozí data vyzvednuta a zobrazena na displej a příznak nových dat je vymazán pro příjem dalších hodnot. Při testování příjmu cyklicky vysílaných hodnot nejsou vysílány požadavky na data od objektů 3-5.

Vysílání požadavků na data je prováděno v obsluze přerušení od čítače/časovače 1 pomocí objektů 3-5, do kterých jsou následně uloženy hodnoty přijaté jako odpověď na požadavek. Tyto hodnoty jsou zobrazeny na displej a přepisují případné hodnoty od objektů 0-3 již zobrazené, ale s větší frekvencí. Ukázka hodnot zobrazených na displeji je na obr. 4.2.

Obr. 4.2: Ukázka zobrazení přijatých hodnot

Komunikace mezi navrhovaným uzlem a testovacím zařízením po CAN sběrnici probíhala podle nastavení a korektně a data byla zobrazována správně. Během testování nebyly pozorovány jakékoliv problémy.

4.2 Ověření sériové komunikace a funkce A/D převodníku

Kontrola správné funkce komunikace po sériové lince RS232C mezi CAN uzlem a programem CAN SETUP běžící na osobním počítači PC s WinXP probíhala prakticky po celou dobu vývoje software. Ve finální verzi odpovídalo nastavení CAN uzlu a jeho následné chování odeslané konfiguraci z PC definované uživatelem. Docházelo sice k chybám přenosu, ale ty byly způsobeny dlouhým propojovacím kabelem a okolním rušením. Chyby byly vždy správně detekovány a protokolem komunikace odstraněny.

Funkce A/D převodníku byla ověřena pomocí jednoduchého teplotního čidla sestaveného z NTC termistoru zapojeného jako napěťový dělič připojený na napětí +5V,

dávající na výstupu napětí 0 až 2,5V při teplotách 0 až 40°C. Toto čidlo bylo připojováno na jednotlivé vstupy A/D převodníku. Digitální hodnota změřeného napětí v rozlišení 8 bitů byla odesílána pomocí funkce *Kontrolní měření* aplikace CAN SETUP do PC a převedena na hodnotu teploty dle obr. 4.3.

T [°C]	R_T [kΩ]	U_{OUT} [V]	8-bit výstup A/D
0	35,5630	0,6163	63d
5	27,1190	0,7784	79d
10	20,8600	0,9667	99d
15	16,2040	1,1790	120d
20	12,6830	1,4138	144d
25	10,0000	1,6667	170d
30	7,9420	1,9317	197d
35	6,3268	2,2072	225d
40	5,0740	2,4816	253d

Obr. 4.3: Hodnoty napětí, teploty a odporu

Teplotní závislost odporu R_T termistoru byla odečtena z aplikačního listu. Hodnota teploty získaná pomocí termistoru byla porovnána v ustáleném stavu s hodnotou teploty z čidla SHT11. Odchylka obou teplot byla do 1°C, kterou vysvětluji rozptylem hodnot odporu rezistoru v děliči a termistoru, nestálostí napájecího napětí 5V a také tolerancí teploty měřené pomocí snímače SHT11, která je ±0,4°C při 25°C. Z tohoto usuzuji na správnou funkci A/D převodníku z hlediska převodu.

5 Závěr

Cílem diplomové bylo navrhnout měřící zařízení komunikující pomocí CAN protokolu. Po seznámení se s tímto standardem, určení měřených veličin a po stanovení základních požadavků jsem jako řídící mikropočítač vybral obvod firmy ATMEL AT89C51cc03 v PLCC44 pouzdru s integrovaným CAN radičem, jako budič CAN sběrnice pak obvod PCA82C250 firmy Philips. Následně jsem provedl návrh zapojení a desky plošných spojů v programu Eagle a zařízení jsem fyzicky realizoval. Provedl jsme vývoj firmware procesoru v jazyce C a konfiguračního programu pro PC v DELPHI 7 a odladil celé zařízení.

Zařízení je schopno komunikovat přenosovou rychlostí až 1MB/s podle standardu CAN 2.0A i CAN 2.0B. Je možno jej napájet ze standardní baterie 9V, nebo síťového DC adaptéru (min. 7V). Odběr proudu se pohybuje okolo 60mA při standardním běhu programu (vysílání dat na CAN sběrnici), což se dá požadovat za hodnotu přijatelnou pro bateriové napájení. Na CAN sběrnici jsou vysílány hodnoty teploty, vlhkosti a rosného bodu z čidla SHT11 a hodnoty 3 napětí v rozsahu 0 až 2,5V přivedené na vstup realizovaného zařízení. Konfigurační software umožňuje nastavení přenosové rychlosti, identifikátorů jednotlivých veličin, volbu a interval odesílaných dat.

Zařízení jsem testoval pomocí komunikace s vývojovým systémem s C167CR v učebně KSI TUL, pro který jsem vytvořil program, který čte data z CAN sběrnice vysílaná realizovaným zařízením rychlostí 100 kb/s a zobrazuje je na displej. Při této komunikaci nebyly pozorovány žádné problémy.

Seznam použité literatury

[1] ROBERT, BOSCH, GmbH: CAN Specification, v 2.0. 1991 [online]. Dostupné z http://www.semiconductors.bosch.de/pdf/can2spec.pdf

[2] HEROUT, P.: *Učebnice jazyka C.* KOOP, 2004. 272 s. ISBN 80-7232-220-6.

[3] HEROUT, P.: *Učebnice jazyka C, 2 díl.* KOOP, 2006. 437 s. ISBN 80-7232-221-4.

[4] CIA [online]. Dostupné z http://www.can-cia.org

[5] HW.CZ [online]. Dostupné z http://www.hw.cz

[6] INFINEON TECHNOLOGIES: *C167CR users manual,* 2000.

[7] INFINEON TECHNOLOGIES: *C167CR instruction set,* 2001.

[8] KEIL SOFTWARE: *Cx51 Compiler Users Guide,* 2000.

[9] ATMEL CORPORATION: *Katalogový list AT89C51cc03*. 2006.

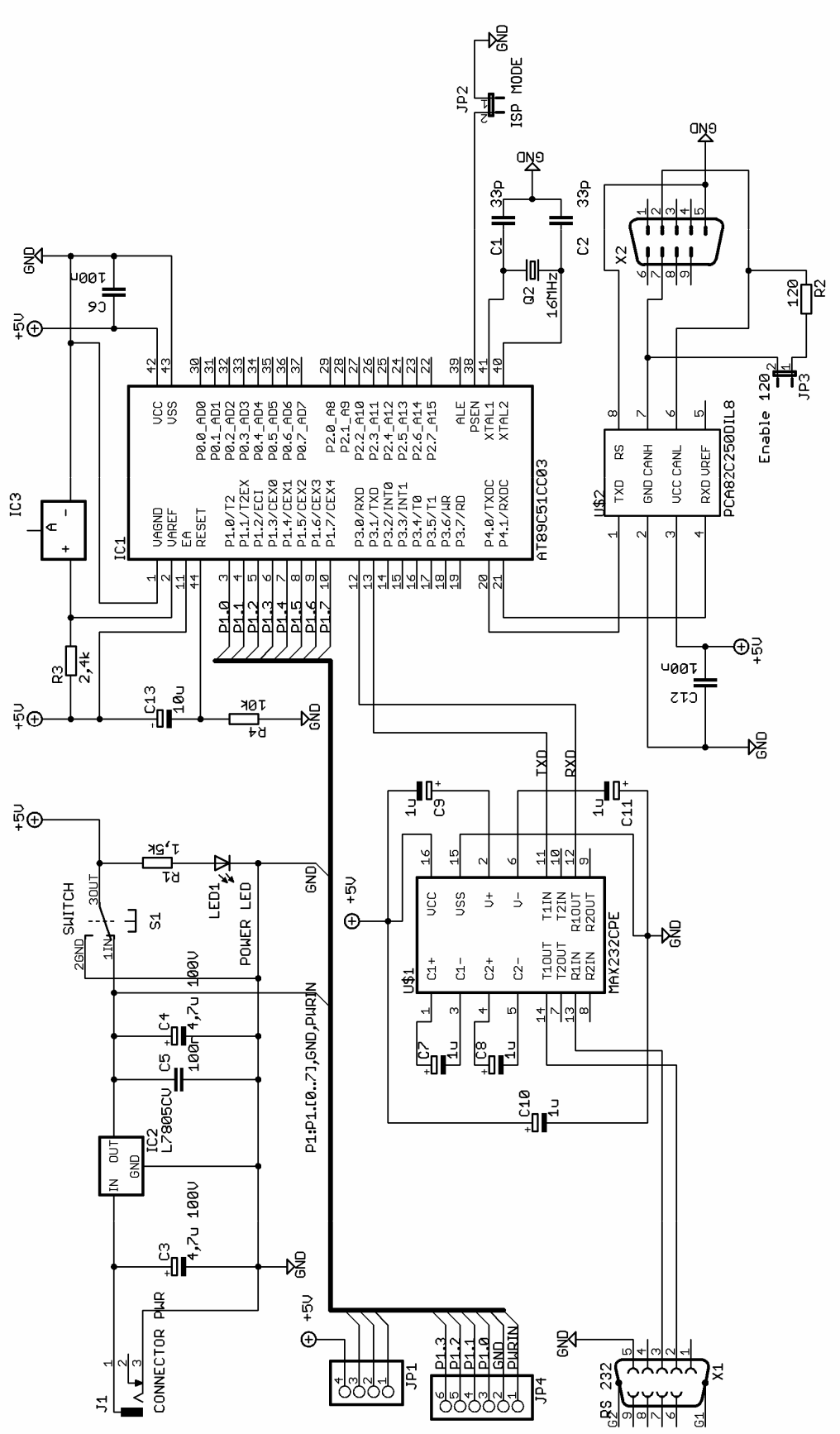
[10] MAXIM INTEGRATED PRODUCTS: *Katalogový list MAX 220-249*. 2004.

[11] PHILIPS SEMICONDUCTORS: *Katalogový list PCA82C250*. 2000.

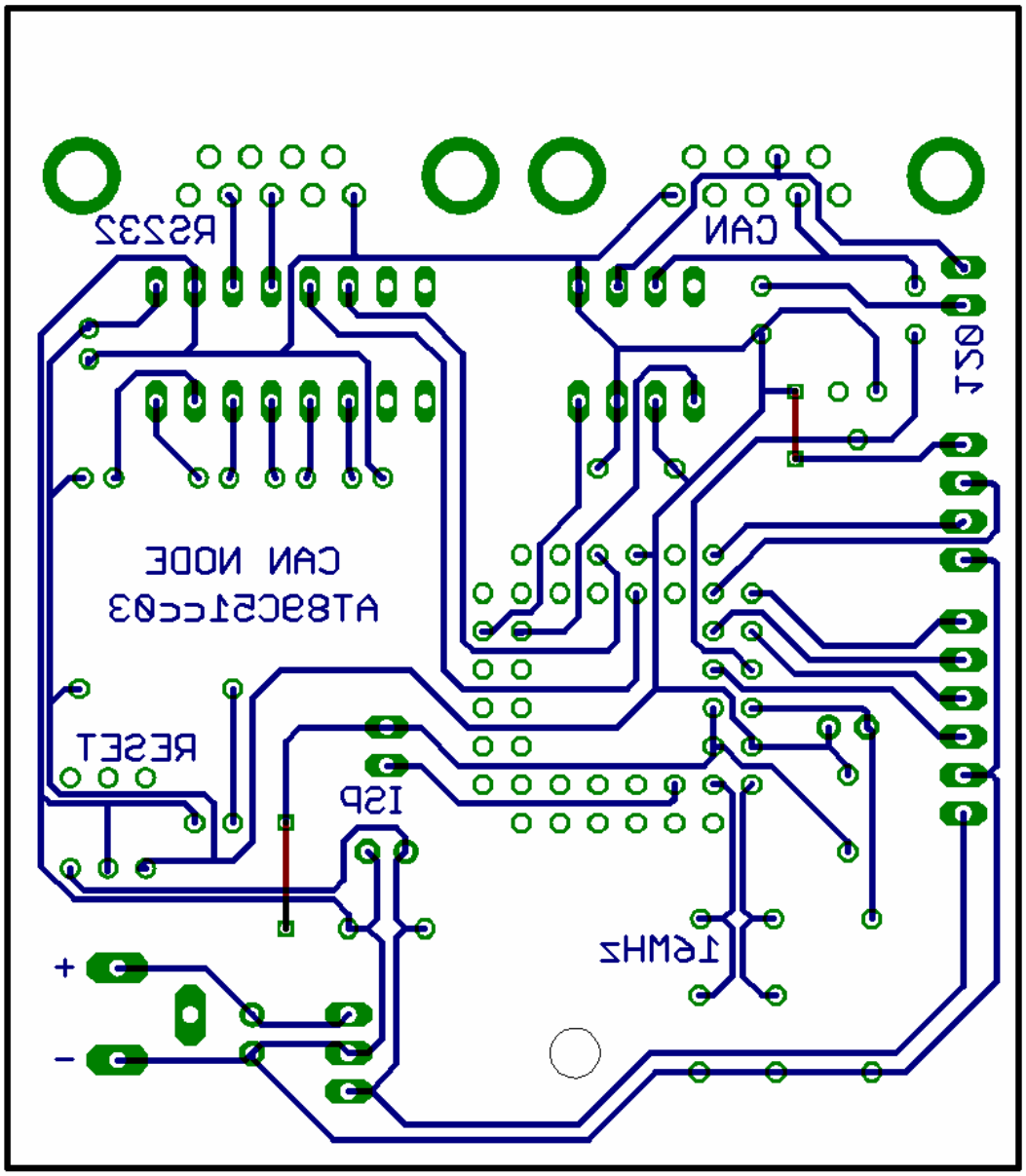
[12] THE SENSIRION COMPANY: *Katalogový list SHT11*. 2004.

Seznam příloh

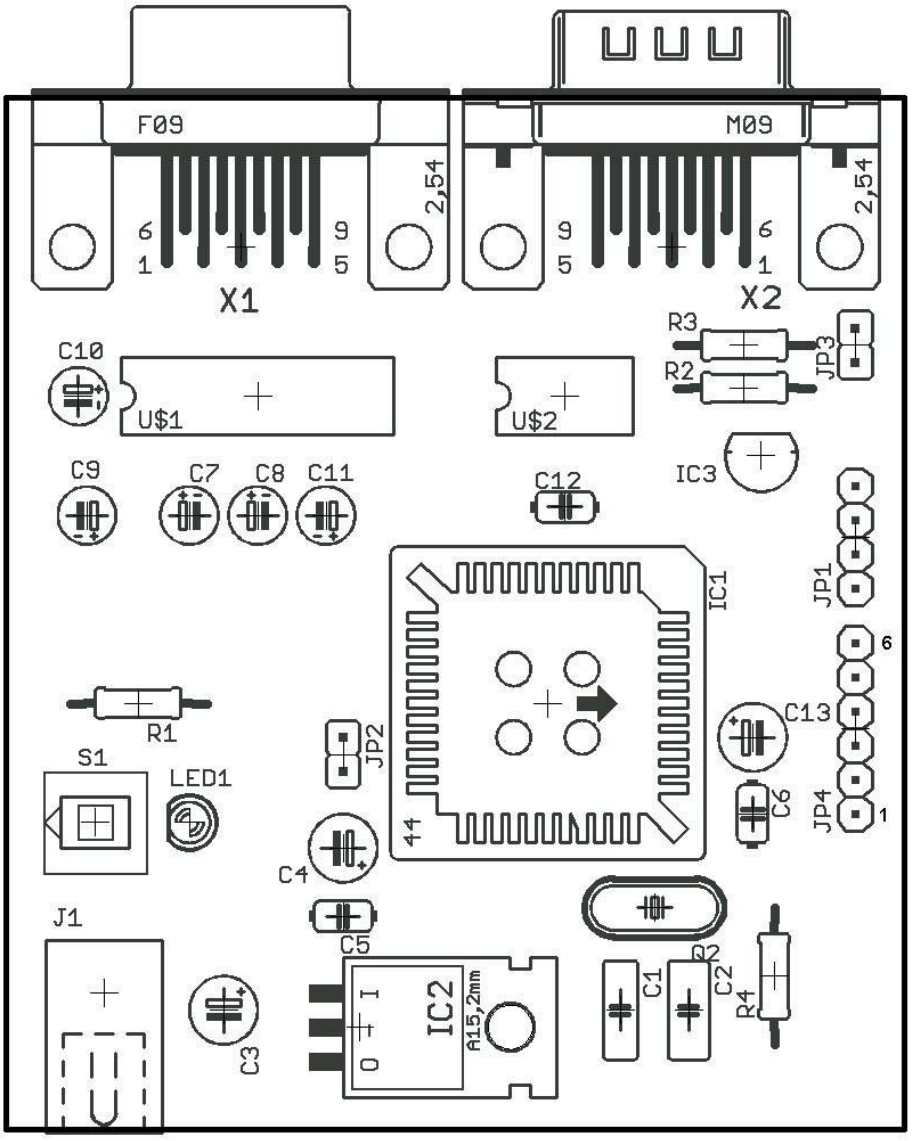
Příloha č.1: Celkové schéma zapojení



Příloha č.2: Vodivý obrazec desky plošných spojů



Příloha č.3: Osazovací výkres a popis



S1 ……. resetovaní tlačítko

LED1 …… signalizace připojeného napětí

J1 …….. konektor pro připojení napájení

X1 …… konektor pro připojení k rozhraní RS232C

X2 …… konektor pro připojení k rozhraní CAN

JP1 ……... konektor pro připojení čidla SHT11

JP2 ……... propojka k uzemnění pinu PSEN (ISP)

JP3 ……..... připojení ukončovacího odporu 120Ω mezi signály CAN_H a CAN_L

JP4 …… vstupní konektor

Příloha č.4: Výpis komunikace mezi CAN uzlem a PC

Odspoda nahoru: Inicializace uzlu, konfigurace uzlu a spuštění aplikace.

... program spuštěn.

Spuštění programu ...

..... Nastavení proběhlo v pořádku .....

Přenos OK, potvrzuji platnost dat.

000000AABAAF Přijato

000000AABAAF Odesílám

Přenos OK, potvrzuji platnost dat.

000001FFAEFF Přijato

000001FFAEFF Odesílám

Přenos OK, potvrzuji platnost dat.

000001EFAEEE Přijato

000001EFAEEE Odesílám

Přenos OK, potvrzuji platnost dat.

000001DFAEDD Přijato

000001DFAEDD Odesílám

Přenos OK, potvrzuji platnost dat.

000001CFABCC Přijato

000001CFABCC Odesílám

Přenos OK, potvrzuji platnost dat.

000001BFABBB Přijato

Chyba přenosu, opakuji.

FC00105FABBB Přijato

000001BFABBB Odesílám

Přenos OK, potvrzuji platnost dat.

000001AFABAA Přijato

000001AFABAA Odesílám

... spojení navázáno.

Inicializace uzlu ...

Komunikační port 1 otevřen

Příloha č.5: Používané registry CAN řadiče

CANPAGE – registr pro výběr aktivního objektu zpráv

7	6	5	4	3	2	1	0
CHNB3	CHNB2	CHNB1	CHNB0	AINC	INDX2	INDX1	INDX0

- INDX0:INDX2 – index datového bajtu 0-7 objektu
- AINC – autoinkrementace indexu datového bajtu
  - 0 – AINC aktivní
  - 1 – AINC neaktivní
- CHNB3:CHNB0 – binární hodnota 0-14 určí aktivní objekt zpráv

CANBT1 - registr časování, slouží k uložení hodnoty BRP.

7	6	5	4	3	2	1	0
-	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	-

CANBT2 - registr časování, slouží k uložení hodnot SJW a PRS.

7	6	5	4	3	2	1	0
-	SJW1	SJW0	-	PRS2	PRS1	PRS0	-

CANBT3 - registr časování, slouží k uložení hodnot PHS1 a PHS2.

7	6	5	4	3	2	1	0
-	PHS2 2	PHS2 1	PHS2 0	PHS1 2	PHS1 1	PHS1 0	-

CANCONCH – konfigurační registr společný všem objektům zpráv. Nastavení je aktivní pro objekt právě vybraný pomocí CANPAGE registru.

7	6	5	4	3	2	1	0
CONCH1	CONCH0	RPLV	IDE	DLC3	DLC2	DLC1	DLC0

- DLC3:DLC0 – délka vysílaných, nebo přijímaných dat (0-8 bajtů)
- IDE – rozlišení identifikátoru
  - 0 – 11 bitová identifikátor dle CAN 2.0A
  - 1 – 29 bitový identifikátor dle CAN 2.0B

- RPLV – automatická odpověď při žádosti o data

  0 – automatická odpověď zakázána

  1 – automatická odpověď povolena

- CONCH1:CONCH0 – konfigurace objektu zpráv

  0 0 – zakázán

  0 1 – start vysílání

  1 0 – povolen příjem

  2 1 – objekt zpráv jako vstupní buffer

CANIE1 – registr povolení přerušení objektů zpráv

7	6	5	4	3	2	1	0
-	IECH 14	IECH 13	IECH 12	IECH 11	IECH 10	IECH 9	IECH 8

Povolení přerušení od příslušného objektu zpráv.

0 – zakázání všech přerušení od objektu zpráv

1 – povolení všech přerušení od objektu zpráv

CANIE0

7	6	5	4	3	2	1	0
IECH 7	IECH 6	IECH 5	IECH 4	IECH 3	IECH 2	IECH 1	IECH 0

Povolení přerušení od příslušného objektu zpráv.

0 – zakázání všech přerušení od objektu zpráv

1 – povolení všech přerušení od objektu zpráv

CANGIE – hlavní registr povolení přerušení

7	6	5	4	3	2	1	0
-	-	ENRX	ENTX	ENERCH	ENBUF	ENERG	-

- ENRX – povolení přerušení při přijmutí zprávy
- ENTX – povolení přerušení při odvysílání zprávy
- ENERCH – povolení přerušení při chybové zprávě
- ENBUF – povolení přerušení buffer
- ENERG – povolení hlavního (general) přerušení

CANSTCH – status registr společný všem objektům zpráv. Nastavení je aktivní pro objekt právě vybraný pomocí CANPAGE registru.

7	6	5	4	3	2	1	0
DLCW	TXOK	RXOK	BERR	SERR	CERR	FERR	AERR

- AERR – příznak chyby odpovědi (nedetekována dominantní úroveň ACK pole)
- FERR – příznak chyby rámce
- CERR – příznak chyby CRC
- SERR – příznak chyby vkládání bitů
- BERR – příznak chyby bitu
- RXOK – příznak úspěšného dokončeného příjmu
- TXOK – příznak úspěšného dokončení vysílání
- DLCW – příchozí zpráva nemá definované pole DLC

CANIDT1-CANIDT4 – registry identifikátoru. Mají odlišné funkce pro 11 bitový a 29 bitový identifikátor.

CANIDT1	7	6	5	4	3	2	1	0
11 bit IDT	IDT10	IDT9	IDT8	IDT7	IDT6	IDT5	IDT4	IDT3
29 bit IDT	IDT28	IDT27	IDT26	IDT25	IDT24	IDT23	IDT22	IDT21

CANIDT2	7	6	5	4	3	2	1	0
11 bit IDT	IDT2	IDT1	IDT0	-	-	-	-	-
29 bit IDT	IDT20	IDT19	IDT18	IDT17	IDT16	IDT15	IDT14	IDT13

CANIDT3	7	6	5	4	3	2	1	0
11 bit IDT	-	-	-	-	-	-	-	-
29 bit IDT	IDT12	IDT11	IDT10	IDT9	IDT8	IDT7	IDT6	IDT5

CANIDT4	7	6	5	4	3	2	1	0
11 bit IDT	-	-	-	-	-	RTRTAG	-	-
29 bit IDT	IDT4	IDT3	IDT2	IDT1	IDT0	RTRTAG	-	-